Apple II



APPLESOFT

Programmieranleitung

Zur Beachtung

Apple Computer Inc. behält sich das Recht von Verbesserungen des in diesem Handbuch beschriebenen Produktes zu jeder Zeit und ohne Ankündigung vor.

AUSSCHLUß JEDER GARANTIE UND HAFTUNG

APPLE COMPUTER INC. GIBT KEINE GARANTIEN, WEDER AUSDRÜCKLICH ODER UNAUSGESPROCHEN, IN BEZUG AUF DIESES HANDBUCH NOCH IN BEZUG AUF DIE IN DIESEM BUCH BESCHRIEBENE SOFTWARE, IHRE QUALITÄT, DURCHFÜHRBARKEIT, VERKÄUFLICHKEIT ODER VERWENDBARKEIT FÜR EINEN BESTIMMTEN ZWECK. APPLE COMPUTER INC. SOFTWARE IST GEKAUFT ODER LIZENSIERT „WIE BESEHEN". DAS RISIKO FÜR DIE QUALITÄT UND DURCHFÜHRBARKEIT LIEGEN GANZ BEIM KÄUFER. SOLLTEN SICH DIE PROGRAMME NACH IHREM KAUF ALS FEHLERHAFT HERAUSSTELLEN, TRÄGT DER KÄUFER (UND NICHT APPLE COMPUTER INC., DEREN DISTRIBUTOR ODER EINZELHÄNDLER) DIE GESAMTEN KOSTEN FÜR ALLE NOTWENDIGEN ARBEITEN, REPARATUREN ODER KORREKTUREN UND JEDWEDE DAVON GEFOLGTEN ODER DARAUS RESULTIERENDEN SCHÄDEN. IN KEINEM FALL IST APPLE COMPUTER INC. HAFTBAR FÜR DIREKTE, INDIREKTE, VERURSACHTE ODER GEFOLGTE SCHÄDEN DIE AUS IRGENDWELCHEN SCHÄDEN AUS DER SOFTWARE RESULTIEREN, SELBST WENN APPLE COMPUTER INC. AUF DIE MÖGLICHKEIT SOLCHER SCHÄDEN HINGEWIESEN WURDE.

Apple Computer International
7, rue de Chartres
92200 Neuilly-sur-Seine
FRANCE

Apple Produkt Nr. A2LD0018

A. Berthmann

Apple II

APPLESOFT

Programmieranleitung



KAPITEL 1
Auf einen guten Anfang

KAPITEL 2
Applesoft für Anfänger

KAPITEL 3
Elementare Programmierung

KAPITEL 4
Alles über Grafiken

KAPITEL 5
Zeichenketten und Tabellen

ANHÄNGE

VORWORT

Dieses Handbuch ist für Leute bestimmt, die die Programmierung in der Applesoft BASIC Sprache erlernen möchten. Mit diesem Buch und einem Apple Computer und ein wenig Zeit und Konzentration ist es gar nicht so schwer, einen Computer zu programmieren. Wie bei jeder neuen Sache werden Sie sich zunächst beim Programmieren unsicher fühlen, aber mit diesem Buch werden Sie dieses Gefühl bald verlieren. Denn es gibt keine Geheimnisse, die man kennen muß, bevor man dieses Buch lesen kann. Alles wird Schritt für Schritt erklärt und mit Beispielen erläutert. Wenn Sie von vorn anfangen, **alle Übungen mitmachen** und bereit sind, ein bißchen Zeit zu opfern, dann garantieren wir Ihnen, daß Sie das Programmieren erlernen werden.

Das eigentliche Geheimnis, wenn es überhaupt eines gibt, liegt darin, daß Sie sich Zeit nehmen müssen und alles ausprobieren. Programmieren erlernt man nicht, indem man dieses oder ein anderes Buch einfach durchliest. Wie Fahrradfahren und Geigespielen kann man es nur durch Übung lernen. Sie müssen Ihre Fehler selbst machen und dann korrigieren, und Sie brauchen sich Ihrer Fehler nicht zu schämen; denn wir alle, Experten oder nicht, machen sie immer wieder.

Wenn Sie allerdings schon programmieren können, wird ein rasches Durchlesen des Buches Sie mit den Eigenschaften des Applesoft vertraut machen. Sie werden wahrscheinlich beeindruckt sein, wie einfach es ist, Grafiken mit hoher Auflösung zu zeichnen und andere Sprachelemente des Applesoft zu benutzen. Und Sie werden feststellen, daß der Apple der richtige Computer ist, um viele verschiedene Aufgaben zu erfüllen.

Dieses Buch ist ein **Lehrbuch.** Zum Nachschlagen von Einzelheiten gibt es das Applesoft BASIC Programmierhandbuch (Best.-Nr. A2L0006). Es enthält ein ausführliches Sachwortregister, eine praktische Karte mit einer Zusammenfassung der Sprache und viele technische Details. Sie sollten es benutzen, **nachdem** Sie dieses Handbuch durchgearbeitet haben. Noch ein Hinweis — wenn Sie das Apple II Basic Programmierhandbuch (Best.-Nr. A2L0005X) gelesen haben, wird Ihnen ein Teil dieses Buches bekannt vorkommen. Das hat seinen guten Grund. Das Konzept is bei unseren Kunden sehr gut angekommen (was uns sehr glücklich macht), so daß wir dieses Handbuch ähnlich abfassen konnten. Wir hoffen, daß das Lesen dieses Buches Ihnen eben so viel Spaß macht, wie uns das Schreiben bereitete.

KAPITEL 1
Auf einen guten Anfang

Einleitung

Wir werden Ihnen zeigen, wie Sie Ihren Apple in Betrieb setzen, und Ihnen helfen, programmieren zu lernen. Wenn Sie ein alter Hase im Programmieren sind, werden Sie im Applesoft BASIC viele neue Möglichkeiten entdecken, mit denen das Programmieren zur Freude wird. Das gilt auch für Anfänger, aber hier eine Warnung : Es ist nicht schwer, programmieren zu lernen, es gelingt jedoch nur durch **Übung.** Wir werden dieses Thema noch häufiger ansprechen; darum merken Sie sich : dieses Buch soll benutzt werden, nicht einfach gelesen.

Wenn Sie Ihren Apple bei einem Händler gekauft haben, wird er gern bereit sein, den Computer schon einmal im Laden aufzubauen, damit es Ihnen auch zuhause gelingt. Aber auch wenn Sie ihn als Geschenk oder per Post bekommen, bereitet es keine Schwierigkeiten, ihn anzuschließen — es ist so einfach, wie das Anschließen einer Stereoanlage, und man braucht dazu kein technisches Verständnis.

Falls Sie es noch nicht getan haben sollten, nehmen Sie sich bitte einige Minuten Zeit, um die Garantiekarte und Besitzurkunde auszufüllen und abzuschicken. Dadurch wird Ihr Apple beim Hersteller registriert, Sie **werden Mitglied der Apple Software Bank** und in die Liste der Apple-Besitzer aufgenommen. Wenn Sie uns die Karte nicht zuschicken, verzichten Sie auf unsere Zeitung, Informationen über neues Zubehör und andere Informationen, die wird den Apple-Besitzern zuschicken.

In diesem Handbuch beschreiben wir einen Apple, der in Applesoft BASIC programmiert werden kann und das eingebaute Autostart ROM besitzt. Wenn Sie ein anderes System haben (z.B. Apple II mit Integer BASIC und das alte Monitor ROM), finden Sie alle nötigen Informationen im Anhang am Schluß des Buches. Wenn sie eine Kassette oder eine Diskette benutzen, dann achten Sie bitte auf die folgenden Symbole :



Durch diese Symbole werden Sie auf Informationen hingewiesen, die besonders für Benutzer von Kassetten und Disketten wichtig sind. Sie sollten die entsprechenden Abschnitte sorgfältig durchlesen, sonst könnte es passieren, daß ein Teil oder das gesamte Applesoft-Programm verloren geht.

Hier ist noch ein wichtiges Symbol :



Dieses Symbol soll Sie auf eine besondere Situation aufmerksam machen. Wenn Sie es nicht beachten, könnten Sie ebenfalls Ihr Programm verlieren.

Was Sie brauchen werden

Das vorliegende deutsche Handbuch ist nicht, wie irrtümlich dargestellt, Bestandteil des Standardlieferumfanges des Apple Computer Systems. Desgleichen sind die Programmkassetten nicht mehr Teil der Lieferung sondern separat erhältlich.

Bildschirmanschluß

Um die Farbmöglichkeiten des Apple II Euro oder Europlus auszunutzen, benötigen Sie in jedem Falle eine PAL-Farbkarte, die in den Einsteckschlitz Nr. 7 eingesetzt wird.

Besitzen Sie einen Farb-Monitor, so können Sie diesen mit einem Verbindungskabel direkt an die PAL-Karte anschließen. Sie können auch einen normalen Farbfernseher betreiben, müssen dan aber zwischen die PAL-Karte und Ihren Fernseher einen HF-Modulator (z.B. SUPER MOD II) schalten. Eine ausführliche Beschreibung der Installation liegt der PAL-Karte bei. Sollten Sie dennoch unsicher sein, wird Ihr Händler gern den Anschluß für Sie vornehmen.

Dieses Handbuch haben Sie im Zubehörkasten gefunden. Der Kasten sollte außerdem folgende dinge enthalten :

1. Das Netzkabel (mit dem Sie Ihr Gerät an das Stromnetz anschließen).

2. Zwei Spielkontrollen (Spielsteuerung) — schwarze Kästchen mit Dreh- und Druckknöpfen, die durch ein Kabel verbunden sind.

3. Ein Kabel, um den Apple mit einem Kassettenrekorder zu verbinden. Dieses Kabel hat zwei Stecker an jedem Ende.

4. Einige Tonbandkassetten. Darauf befinden sich Programme für den Apple.

Außer dem Apple selbst und dem Inhalt des Zubehörkastens werden Sie noch zwei Dinge brauchen, die im Folgenden beschrieben werden :

1. **Sie brauchen eines der folgenden Dinge** (beide zu besitzen ist zwar nützlich, aber nicht unbedingt notwendig) :

   a. Einen Kassettenrekorder.

ODER

   b. Ein Apple Floppy Laufwerk (Disk II) mit Controller Karte.

2. **Zusätzlich werden Sie eines der folgenden Dinge brauchen :**

   a. Einen Schwarzweiß- oder Farbmonitor mit einem Verbindungskabel, das an die Videobuchse des Apple II paßt. Ihr Händler wird es Ihnen besorgen können.

ODER

   b. Einen normalen Schwarzweiß-Fernseher und einen HF-Modulator mit Verbindungskabeln. Dieser Modulator ändert das Ausgangssignal des Apple so, daß Ihr Fernseher es in ein Bild umwandeln kann. Es gibt viele verschiedene Modulatoren. Speziell für den Apple wurde der sogen. SUPERMOD II entwickelt, den Sie bei Ihrem Computerhändler erhalten können.

ODER

   c. Einen normalen Farbfernseher im PAL-System und eine PAL-Karte, die in den Einsteckschlitz 7 in Ihrem Apple eingesetzt wird. Auch hier brauchen Sie zusätzlich noch ein Verbindungskabel.

Mit jedem Modulator erhalten Sie eine ausführliche Beschreibung, damit Sie ihn anschließen können. Das normale Fernsehprogramm kann trotzdem weiterhin empfangen werden.

Ein Schwarzweiß-Monitor oder -Fernseher reicht völlig aus, nur verzichten Sie damit auf einen wesentlichen Vorteil des Apple. Die Farbbilder, die er erzeugt, erscheinen dann in verschieden abgestuften Grautönen.

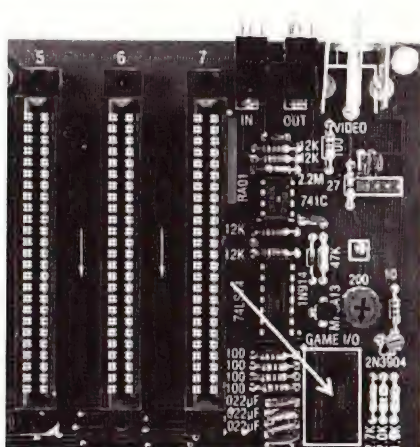
Anschluß des Fernsehers

Ein Schwarzweiß- oder Farbmonitor wird angeschlossen, indem man einfach den Stecker VIDEO OUT (auf der Rückseite des Apple) und den Eingangsstecker des Monitors mit dem entsprechenden Kabel verbindet.

Wenn Sie ein normales Fernsehgerät anschließen wollen, benötigen Sie einen HF-Farbmodulator. Um Farbbilder zu erhalten, benötigen Sie in beiden Fällen eine Pal-Karte. Öffnen Sie den Apple, indem Sie mit beiden Händen den Gehäusedeckel hinten gerade hochziehen. Der Einbau des Modulators wird in der mitgelieferten Beschreibung erklärt.

Anschluß der Spielkontrollen

Wenn Sie den Gerätedeckel geöffnet haben, sehen Sie rechts in der hinteren Ecke einen Steckkontakt, der mit GAME I/O beschriftet ist. Stecken Sie dort den Stecker der Spielkontrolle vorsichtig hinein, da er sehr fein ist, und achten Sie darauf, daß auch alle Stifte eingeführt sind. Dabei muß der weiße Punkt auf dem Stecker nach vorn (zur Tastatur hin) weisen.



Das Floppy-Laufwerk Disk II

Das Floppy-Laufwerk muß ganz vorsichtig ausgepackt werden. Der Anschluß des Gerätes wird in dem mitgelieferten DOS-Handbuch (Disk Operating System) ausführlich erklärt. Lesen sie dazu zunächst das Vorwort und das erste Kapitel (Seite 2 bis 8) in diesem Handbuch durch.

Der Kassettenrekorder

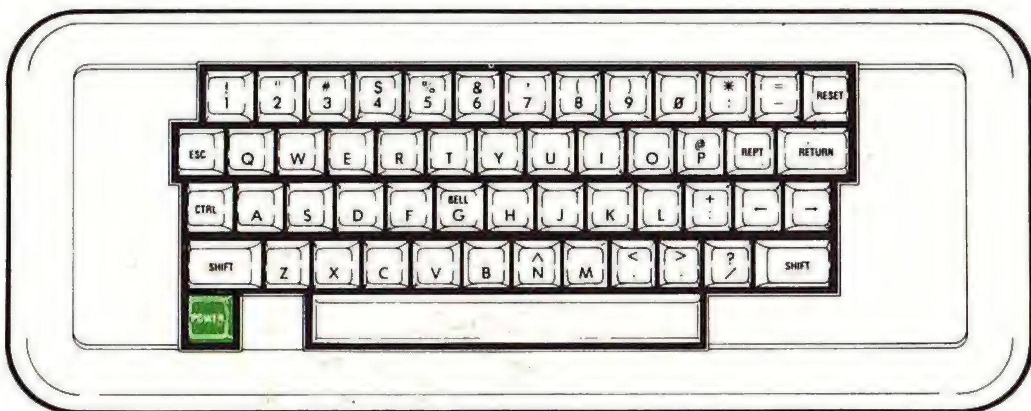
(Falls Sie kein Floppy-Laufwerk benutzen oder falls Sie sowohl ein Laufwerk als auch einen Rekorder benutzen wollen.)

Um Ihren Kassettenrekorder mit dem Apple zu verbinden, benötigen Sie das mitgelieferte Kabel, das zwei Stecker an jedem Ende hat. Ein schwarzer Stecker wird an die Buchse MIC oder MICROPHONE des Rekorders angeschlossen, der schwarze Stecker am anderen Ende des Kabels hinten am Apple bei der Markierung CASSETTE OUT. Nun werden die grauen Stecker angeschlossen, beim Kassettenrekorder an die Buchse EARPHONE oder MONITOR und beim Apple an der Markierung CASSETTE IN. „OUT" bedeutet hier Ausgabe vom Computer an den Rekorder, „IN" Eingabe vom Rekorder in den Computer. Jetzt brauchen Sie den Kassettenrekorder nur noch ans Stromnetz anzuschließen, und er ist startklar.

Zum Schluß schließen Sie den Gehäusedeckel des Apple und verbinden den Computer mit dem Stromnetz (der Anschluß für das Netzkabel befindet sich hinten am Apple neben dem EIN/AUS-Schalter). Ihr Apple ist nun vollständig aufgebaut, und Sie können beginnen, die faszinierende Welt des Computers zu erforschen.

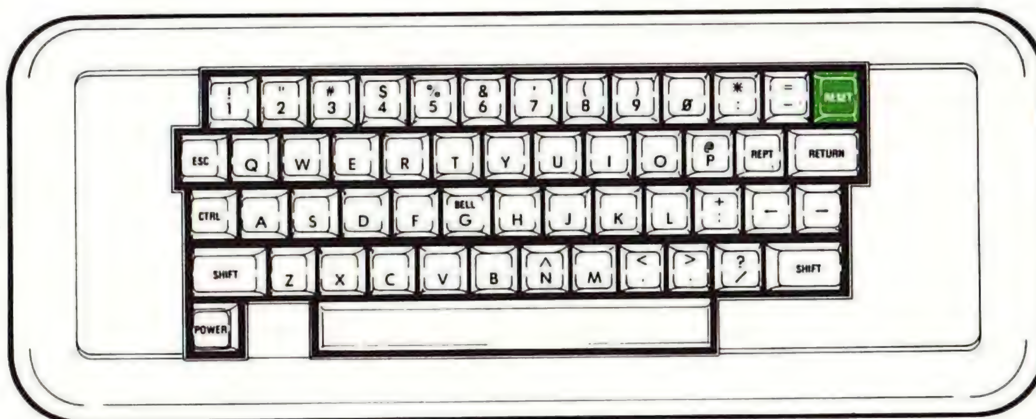
Die Tastatur des Apple

Wenn Sie den Apple vollständig aufgebaut haben, können Sie ihn einschalten. Der EIN/AUS-Schalter befindet sich hinten am Gerät neben dem Netzanschluß. Drücken Sie ihn nach oben. Zur Belohnung wird die POWER-Lampe unten in der Tastatur aufleuchten. Diese sieht wie eine Taste aus; es ist aber keine, und sie kann nicht gedrückt werden. Außerdem sollte jetzt die Zeile „APPLE II' oben auf dem Bildschirm erscheinen. Darunter (am linken Rand) werden Sie eine eckige Klammer (]) und ein blinkendes Quadrat, den sogen. „Cursor" sehen.





Wenn Sie dieses Bild nicht auf Ihrem Bildschirm sehen, machen Sie sich keine Sorgen. Haben Sie die Applesoft-Firmware-Karte (Best.-Nr. A2B0009X) in den Apple eingesetzt, dann überprüfen Sie, ob der Schalter an der Rückseite der Karte nach oben zeigt. Nun schalten Sie Ihren Apple aus und wieder an. Falls Ihr Bildschirm immer noch von der Beschreibung abweicht oder falls Sie nicht Applesoft auf der Karte haben, drücken Sie die Taste RESET oben rechts auf der Tastatur. Beim Loslassen der RESET -Taste wird Ihr Apple einen Piepton abgeben.

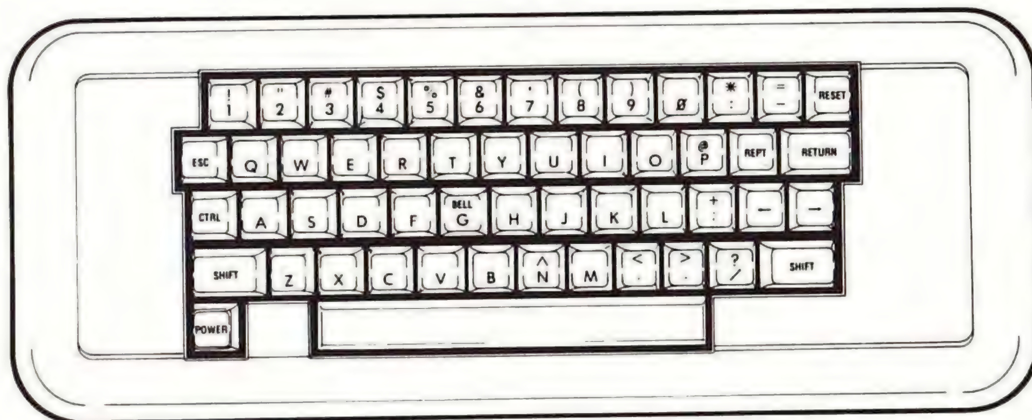


Ein Druck auf die RESET -Taste hilft Ihnen in den meisten Fällen, wenn der Apple nicht richtig auf Ihre Befehle reagiert. (Welche Reaktionen zu erwarten sind, werden Sie beim Durcharbeiten dieses Handbuches erfahren). Falls das immer noch nicht hilft, so wird das Problem vermutlich durch Aus- und wieder Einschalten des Gerätes gelöst.

Wenn Sie ein Floppy Laufwerk benutzen, dann wird beim Einschalten des Apple folgendes passieren: Zunächst werden Sie ein Klicken hören, danach ein summendes Geräusch, und die rote Birne „IN USE" leuchtet auf. Das Laufwerk wird summen und summen, als ob es nie wieder aufhören wollte. Und richtig, es wird solange summen, bis Sie die RESET -Taste drücken. Tun Sie es! Dann wird die Zeile „APPLE II" vom Bildschirm verschwinden, und das Bereitschaftszeichen und der Cursor werden unten rechts erscheinen.



Betrachten Sie bitte nun die Tastatur des Apple. Sie ist etwas anders als die einer Schreibmaschine. Hier gibt es u.a. keine kleinen Buchstaben, sondern nur große. Aber mehr brauchen Sie auch nicht, um Programme in Applesoft BASIC zu schreiben.



Auf dem Diagram sehen Sie, daß wir zwei SHIFT -Tasten haben. Mit Hilfe der SHIFT -Tasten können wir die Zahl der Zeichen im Verhältnis zum Tastenvorrat nahezu verdoppeln. Wenn wir für jedes Zeichen eine besondere Taste hätten, wäre die Tastatur viel zu groß und unübersichtlich.

Wenn man eine Taste mit zwei Zeichen drückt, erscheint das untere auf dem Bildschirm. Drückt man jedoch die Taste zugleich mit einer der beiden SHIFT -Tasten, so erscheint das obere Zeichen. Dazu zwei Beispiele : SHIFT und Kommataste gleichzeitig gedrückt ergeben das Zeichen <, SHIFT und Punkttaste gleichzeitig gedrückt, ergeben das Zeichen >. Auf der Appletastatur befinden sich noch weitere Symbole, die auf einer normalen Schreibmaschine nicht vorhanden sind. Probieren Sie ruhig einige Zeichen aus!

Ist eine Taste nur mit einem Zeichen markiert, so hat ein gleichzeitiges Drücken der SHIFT -Taste keine Wirkung, bis auf zwei Ausnahmen : bei der M -Taste und bei der G -Taste.



Mit SHIFT-M erhält man eine eckige Klammer, rechts geschlossen (]). Bei der G -Taste steht das Wort „BELL" über dem Buchstaben. Deshalb erscheint beim Drücken von SHIFT-G aber nicht etwa eine Glocke auf dem Bildschirm, sondern ein einfaches „G". Die Bedeutung des Wortes „BELL" auf der G -Taste werden wir Ihnen später erklären.

Ein wesentlicher Unterschied zwischen der Apple- und Schreibmaschinentastatur besteht darin, daß das kleine „L" nicht als Zahl Eins „1" verwendet werden kann, da der Apple keine kleinen Buchstaben kennt. Wenn Sie also die Angewohnheit haben, ein kleines „L" als „1" zu schreiben, müssen Sie sich hier umstellen.

Als die Mathematiker der Hindus vor vielen Jahren einen Kreis als Darstellung für die Zahl Null einführten, kannten sie das lateinische Alphabet noch nicht. Sie wählten ein Symbol, das in ihrem Alphabet nicht vorkam, das aber bei uns wie der Buchstabe „O" aussieht. Ein Computer (und jeder, der logish denkt) muß abr Null und „O" unterscheiden können. Zu diesem Zweck versieht der Apple, wie viele andere Computer auch, die Null mit einem Schrägstrich : Ø. Daher gibt es auf der Tastatur des Apple getrennte Tasten für die Null (Ø) und den Buchstaben (O). Probieren Sie diese Tasten einmal aus!



Wenn Sie ein wenig auf der Tastatur herumspielen, füllt sich der Bildschirm langsam. Sie können ihn wieder löschen, indem Sie die Taste **ESC** drücken. **ESC** ist eine Abkürzung für ESCape und bedeutet etwa „entweichen". Die **ESC** -Taste löst kein Zeichen auf dem Bildschirm aus.

Drücken Sie die **ESC** -Taste und danach den „Klammeraffen". Das ist das merkwürdige Zeichen @ über dem Buchstaben „P". Sie müssen also **SHIFT** und diese Taste gleichzeitig drücken. Die **ESC** -Taste wird im Gegensatz zur **SHIFT** -Taste nicht gleichzeitig mit anderen Tasten gedrückt. Um den Bildschirm zu löschen, müssen Sie also insgesamt drei Tasten betätigen : zuerst **ESC** und dann **SHIFT-P** gleichzeitig.



Die Darstellung von Tastenkombinationen

In diesem Abschnitt wollen wir Ihnen zeigen, wie wir verschiedene Tastenkombinationen in diesem Buch darstellen werden.

Wenn eine bestimmte Taste gedrückt werden soll, wird das Zeichen auf der Taste dargestellt. Soll z.B. die Taste mit dem Buchstaben „H" gedrückt werden, so erscheint im Text das Zeichen dieser Taste, H. Um eine Folge von Tasten darzustellen, geben wir die Zeichen in der Reihenfolge an, in der die entsprechenden Tasten nacheinander gedrückt werden sollen, z.B.

H E L L O

Gelegentlich müssen Sie zwei Tasten gleichzeitig drücken. Um das Dollarzeichen ($) zu erhalten, muß man z.B. gleichzeitig die SHIFT -Taste und die 4 -Taste drücken. In solchen Fällen werden wir beide Zeichen übereinandersetzen :



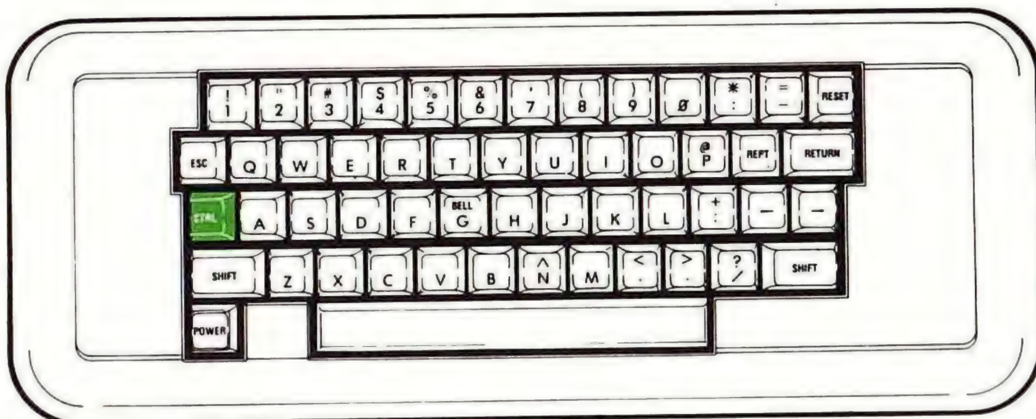
Die obere Taste wird niedergehalten, während die untere gedrückt wird. Hier zeigen wir Ihnen in unserer neuen Schreibweise, wie man den Bildschirm löscht :



Probieren Sie es noch einmal!

Control und andere merkwürdige Zeichen

Wenn Sie die Taste 5 drücken, erscheint die Zahl 5 auf dem Bildschirm. Sie werden das wahrscheinlich auch so glauben, aber probieren Sie es trotzdem aus. Wenn Sie gleichzeitig die SHIFT -Taste drücken, erscheint das Prozentzeichen (%) auf dem Bildschirm. Stimmt's? Durch die SHIFT -Taste bekommen die Tasten zwei verschiedene Funktionen. Es gibt sogar einige Tasten, die drei Funktionen haben. Diese dritte Funktion erhält man, indem man gleichzeitig mit der entsprechenden Taste die CTRL -Taste drückt. ,,CTRL'' ist eine Abkürzung für ,,ConTRol'' und bedeutet Kontrolle. Mit Hilfe der CTRL -Taste erscheinen jedoch keine neuen Zeichen auf dem Bildschirm, im Gegenteil, die CTRL -Eingaben erscheinen überhaupt nicht auf dem Bildschirm. Vielmehr wird der Computer dadurch veranlaßt, bestimmte Dinge zu tun.



Drücken Sie einmal gleichzeitig die CTRL - und G -Taste :



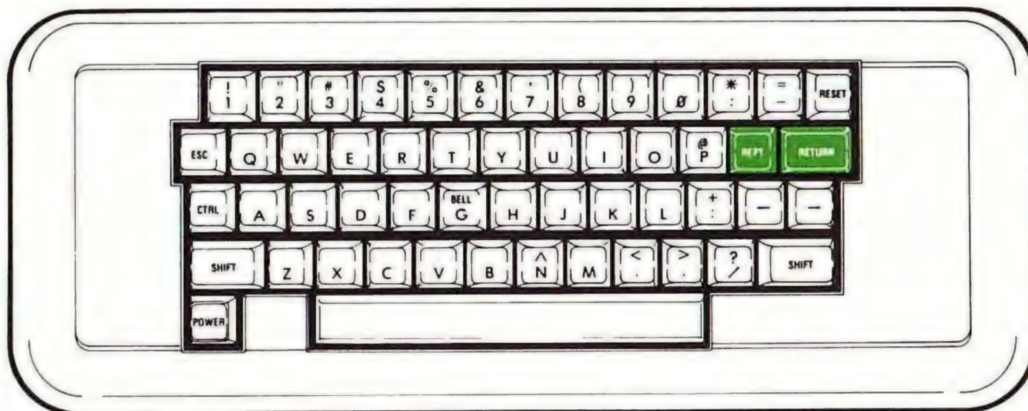
Nun hört man nicht etwa ein Glockengeläute (BELL heißt Glocke), sondern der Computer gibt einen Piepton von sich. Dieser Piepton ertönt immer dann, wenn der Computer Sie auf irgendetwas aufmerksam machen möchte. Die Kombination CTRL-G wird nur aus historischen Gründen „BELL" genannt, weil diese Tastatur aus der eines Fernschreibers entwickelt wurde. Und dort läutete bei der Kombination CTRL-G ursprünglich tatsächlich ein Glöckchen auf.

Eine weitere Taste, die Sie normalerweise nicht auf einer Schreibmaschine finden, ist die REPT -Taste. REPT ist die Abkürzung für „REPeaT" und bedeutet Wiederholung. Wenn Sie diese Taste gleichzeitig mit einer anderen drücken, dann wird das Zeichen der anderen Taste so lange getippt, bis Sie die REPT -Taste loslassen. Drücken Sie **zuerst** die Taste, die wiederholt werden soll, und **dann** die REPT -Taste. Probieren sie es selbst aus!

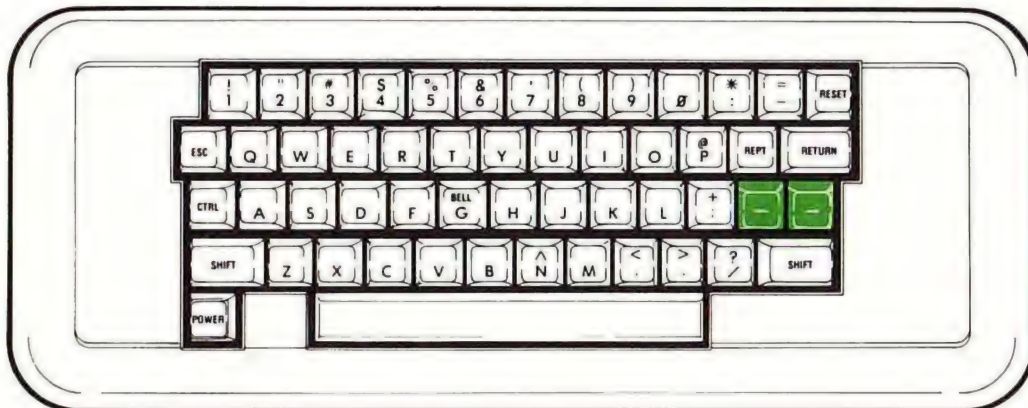
Es gibt auch eine mit RETURN markierte Taste. Bei älteren Maschinen war dies die „Wagenrücklauf"-Taste. Beim Apple bewirkt ein Drücken dieser Taste, daß der blinkende Cursor auf dem Bildschirm eine Zeile tiefer wieder an den linken Rand zurückkehrt. Die RETURN -Taste übermittelt dem Computer außerdem noch eine besondere Nachricht, die jedoch erst später im Buch erläutert werden soll. Es kann zuweilen vorkommen, daß der Computer nach dem Drücken der RETURN -Taste „Piep" sagt und auf dem Bildschirm die Nachricht

```
?SYNTAX ERROR
```

erscheint. Wir werden diese Reaktion später erklären, zunächst können Sie die Sache ignorieren.



Zwei Tasten haben wir noch nicht besprochen, die ← und → Tasten. Mit ihnen wird der Cursor nach links oder nach rechts bewegt. Einzelheiten dazu werden später noch erläutert. Probieren Sie diese und alle anderen Tasten immer wieder aus. Keine Angst, beim Tippen können Sie den Computer nicht beschädigen, es sei denn, Sie benutzen dazu einen Hammer. Also, probieren Sie alles aus! Spielen Sie mit Ihren Fingern!



Die Einstellung des Kassettenrekorders

(Wenn Sie keinen Kassettenrekorder benutzen, können Sie diesen Abschnitt überschlagen und bei dem Thema „BENUTZUNG EINES FLOPPY LAUFWERKS" weiterlesen.)

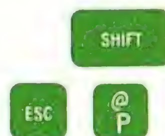
Drücken Sie die **RETURN**-Taste. Am linken Rand des Bildschirms erscheint die eckige rechte Klammer und der blinkende Cursor. Daran sehen Sie, daß Sie „im Applesoft sind" oder daß Sie „Applesoft geladen haben" (wie man sagt). Jetzt können Sie die Lautstärke an Ihrem Kassettenrekorder einstellen.

Wenn Sie eine Kassette abspielen, möchten Sie doch meistens hören, was darauf ist. Wenn das Gerät zu leise eingestellt ist, entgehen Ihnen einige Worte bzw. Klänge. Wenn es zu laut ist, ist es störend.

Wenn Sie eine Kassette für den Apple abspielen, soll die Information ja dem Computer übermittelt werden. Wenn die Lautstärke also zu leise eingestellt ist, wird er manches nicht mitkriegen und sich mit einer Fehlermeldung beschweren. Er wird sich auch beschweren, wenn das Gerät zu laut eingestellt ist.

Um die richtige Lautstärke zu finden, müssen Sie ein wenig experimentieren. Sie werden die Lautstärke zunächst relativ leise einstellen und prüfen, ob der Computer die Information richtig aufnimmt. Wenn nicht, vergrößern Sie die Lautstärke ein wenig und probieren es erneut u.s.w. Irgendwann werden Sie die richtige Lautstärke gefunden haben, und der Apple wird es Ihnen mit einem „Piepton" anzeigen.

Bevor Sie mit der Einstellung beginnen, löschen Sie den Bildschirm mit



Legen Sie nun die Kassette mit der Aufschrift COLOR DEMOSOFT in den Rekorder ein. Jedesmal, wenn Sie die Lautstärke überprüfen wollen, müssen Sie die folgenden Schritte ausführen :

1. Spulen Sie die Kassette zum Anfang zurück.

2. Drücken Sie die Abspieltaste.

3. Tippen Sie folgendes ein :



Wenn Sie dies getan haben, verschwindet der Cursor vom Bildschirm. Es kann bis zu 15 Sekunden dauern, bis etwas passiert. Dabei gibt es die folgenden Möglichkeiten :

a. Die Nachricht ?SYNTAX ERROR erscheint.

b. Es passiert überhaupt nichts.

c. Die Nachricht ERR oder ERROR erscheint (mit oder ohne Piepton).

d. Der Computer sagt „Piep", und auf dem Bildschirm erscheint nichts.

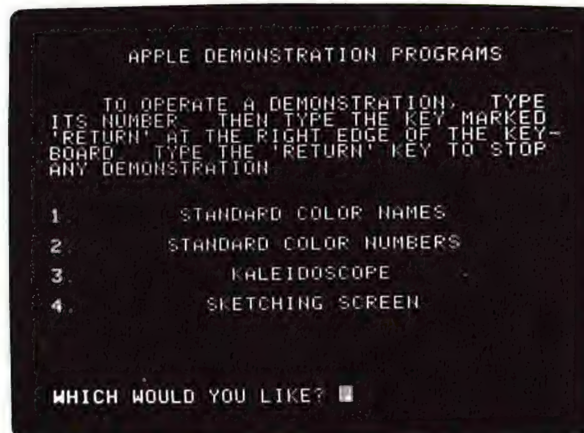
Im Falle a ändern Sie die Lautstärke nicht, sondern beginnen wieder mit Schritt 1 und spulen die Kassette zurück.

In den Fällen b und c warten Sie bitte mindestens 15 Sekunden, bevor Sie aufgeben. Wenn danach kein Bereitschaftszeichen oder der Cursor erscheint und der Apple auch sonst nicht auf Tastendruck reagiert, drücken Sie die RESET -Taste, stellen die Lautstärke ein wenig höher ein und fangen wieder mit Schritt 1 an. In ganz seltenen Fällen kann es einmal vorkommen, daß der LOAD Befehl nicht richtig angenommen wird, d.h. daß der Cursor sofort auf dem Bildschirm erscheint, ohne abzuwarten, bis die Kassette gelesen ist. Falls das passieren sollte, schalten Sie Ihren Apple einfach aus und danach wieder an (der EIN/AUS-Schalter befindet sich hinten am Gerät), und versuchen Sie es noch einmal.

Im Falle d sind Sie auf der richtigen Spur. Wenn Sie den Piepton hören, warten Sie noch einmal 15 Sekunden. Dann werden Sie entweder eine Fehlermeldung bekommen (Fall c), oder das Bereitschaftszeichen (]) und der blinkende Cursor erscheinen wieder auf dem Bildschirm. Wenn das geschieht, spulen Sie die Kassette zurück, und markieren Sie sich diese Lautstärkeneinstellung am Kassettenrekorder. Sie können diese Einstellung dann immer benutzen, wenn Sie wieder eine Kassette einlesen möchte. Tippen Sie nun die Tastenfolge.



Ihr Bildschirm sollte danach folgendermaßen aussehen :



Die übliche Methode, Kassetten einzulesen

(Sie sollten die richtige Lautstärke eingestellt haben)

1. Sie spulen die Kassetten zurück.

2. Sie drücken die Abspieltaste.

3. Sie schreiben LOAD.

Wenn Sie RETURN drücken, verschwindet zunächst der Cursor. Während etwa 15 bis 20 Sekunden passiert nichts. Danach hören Sie einen Piepton. Das bedeutet, daß die auf dem Band befindliche Information jetzt in den Computer eingelesen wird. Nach einer weiteren Zeitspanne, deren Länge von der Länge der einzulesenden Information abhängt, hören Sie erneut den Piepton, und Bereitschaftszeichen und Cursor erscheinen wieder. Diese Zeitspanne ist selten länger als ein paar Minuten. Nun machen Sie noch folgendes.

4. Halten Sie das Tonband an und spulen Sie das band zurück. Die Information ist in die Maschine eingelesen worden, und Sie brauchen das Tonbandgerät nun nicht mehr.

5. Gelben Sie RUN ein, und drücken Sie dann **RETURN** ! Nun wird das eingelesene Programm ausgeführt.



Wenn Sie Ihren Apple in der Applesoft Programmiersprache benutzen, muß auch die vom Band eingelesene Information in Applesoft geschrieben worden sein. Wenn Sie etwas laden, das in einer anderen Computersprache geschrieben ist, kann dabei Unvorhersagbares herauskommen. Merkwürdige Fehlermeldungen und eigenartige Zeichen auf dem Bildschirm können erscheinen, möglicherweise reagiert der Apple nicht mehr auf die Tastatur, und alles Mögliche. Wenn Ihnen so etwas passiert, können Sie einfach Ihr Gerät aus- und dann wieder einschalten. Damit sollte alles wieder normal werden.

Computerfritzen gebrauchen verschiedene Ausdrücke, um den Vorgang des Einlesens vom Tonband in den Computer zu beschreiben. Man sagt etwa, der Computer „liest" das Band. Oder, er „lädt" den Bandinhalt. Die Information auf dem Band wird „geladen". Alle diese Ausdrücke beziehen sich auf denselben eben beschriebenen Vorgang.

Ein nützlicher Hinweis

Was findet der Computer an diesen Tonbändern wohl so interessant? Hören Sie sich doch eins an -- es klingt ja nicht gerade wie Musik! Trotzdem können Sie erkennen, auf welche Geräusche der Computer achtet. Zunächst ist anfänglich ein stetiger Ton, dann ein „Blip", und dann wieder der stetige Ton. Dieser Ton hat eine Frequenz von genau 1000 Hertz. Das ist fast genau zwei Oktaven über dem mittleren C. Und nach diesem anfänglichen Ton kommt dann eine Folge von Tönen, die sich fast so wie starker Regen anhören.

Wenn Sie die Qualität einer Tonbandaufnahme gut beurteilen können, werden Sie schnell feststellen, ob Sie ein Computer Tonband vor sich haben. Wenn Sie durch bloßes Hinhören verstehen, was auf einem Computerband geschrieben ist, haben Sie ein ungewöhnliches Gehirn, und Sie werden es unter Computern wahrscheinlich weit bringen.

Wie man das Floppy Laufwerk benutzt

(Überschlagen Sie diesen Abschnitt, falls Sie kein Floppy Gerät DISK II benutzten.)

Ein Floppy Laufwerk für Disketten ist viel schneller als ein Tonbandgerät und auch viel einfacher in der Benutzung. Andererseits ist das Gerät nicht so robust, und Sie müssen es sorgsam behandeln. Auf den Seiten 5 und 6 des DOS Handbuches finden Sie viele Hinweise, wie Sie mit Ihrem Gerät umgehen sollten. Lesen Sie sich die entsprechenden Abschnitte auf jenen Seiten genau durch (falls Sie es noch nicht getan haben sollten).

Der letzte Abschnitt des ersten Kapitels des DOS Handbuches bespricht, wie man die Disketten einlegt und wieder herausnimmt. Nehmen sie jetzt die Diskette mit der Aufschrift SYSTEM MASTER DISK aus ihrer Verpackung, und schieben Sie sie in das Gerät mit der Beschriftung nach oben und dem ovalen Ausschnitt zur Rückseite des Gerätes hin, so wie es auch im DOS Handbuch beschrieben ist.

Disketten sind deshalb so angenehmen, weil es so leicht fällt, verschiedene Gruppen von Information zu speichern und einzulesen. Jede Gruppe von Information wird **Datei** genannt und wird unter einem bestimmten Namen abgespeichert. Wenn z.B. ein Programm Adressen verwaltet, könnte man es auf der Diskette als Datei „ADRESSEN" speichern.

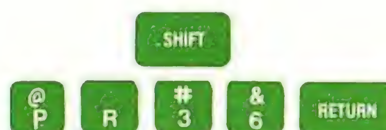
Das DOS Betriebssystem besteht aus einer Anzahl von Programmen, die dafür verantwortlich sind, wo auf der Diskette gewisse Dateien zu finden sind, wo eine neue Datei hingeschrieben werden soll, u.s.w. Man kann diese Betriebssystemprogramme als zusätzliche Fähigkeiten dem Applesoft angliedern. Dieser Vorgang wird „das System booten" oder „das System starten" genannt.

Man kann nun das System auf mehrere Arten starten. Eine Methode besteht darin, den Apple einfach aus- und dann einzuschalten. Daraufhin wird die „IN USE" Birne am Gerät erneut aufleuchten, die vorhin beschriebenen Geräusche werden wieder zu hören sein, geradeso wie beim ersten Einschalten Ihres Computers. Diesmal wird der summende Ton des Geräts von selbst aufhören. Wenn das geschieht und die „IN USE" Birne am Floppy Laufwerk wieder ausgeht, müsste der Titel „APPLE II" auch vom Bildschirm verschwinden. Sie sollten jetzt folgendes auf dem Bildschirm sehen :

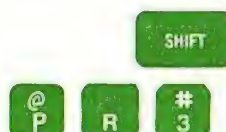


Wenn diese Nachricht erscheint, wissen Sie, daß DOS nun gestartet worden ist. Die rechteckige Klammer ist das Bereitschaftszeichen, an dem Sie sehen, daß Applesoft geladen ist, d.h. der Computer versteht jetzt Applesoft und wartet auf Befehle.

Hier ist eine andere Methode, DOS zu starten :



Sie schreiben das auf Ihrem Apple. Falls die Controller Karte nicht im Einsteckschlitz 6 eingesteckt ist, schreiben Sie

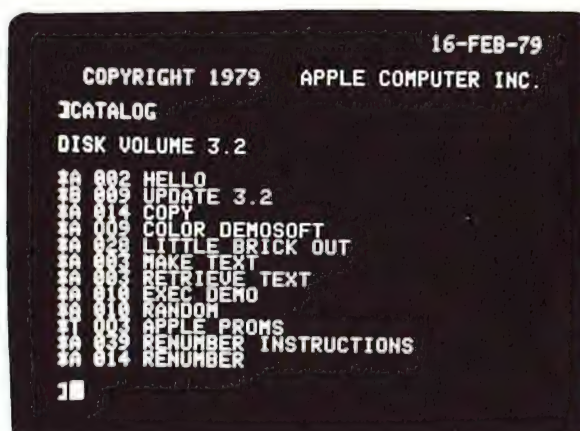


gefolgt von der Nummer des Einsteckschlitzes, in dem die Controller Karte des Floppy Laufwerks eingesteckt ist. Sie drücken nun die **RETURN** -Taste.

Die Diskette mit der Aufschrift „SYSTEM MASTER" ist eine besondere Diskette, denn auf ihr befinden sich viele nützliche Programme, die wir später gebrauchen werden. Um den Inhalt dieser Diskette zu sehen, schreiben Sie den CATALOG Befehl, also

CATALOG **RETURN**

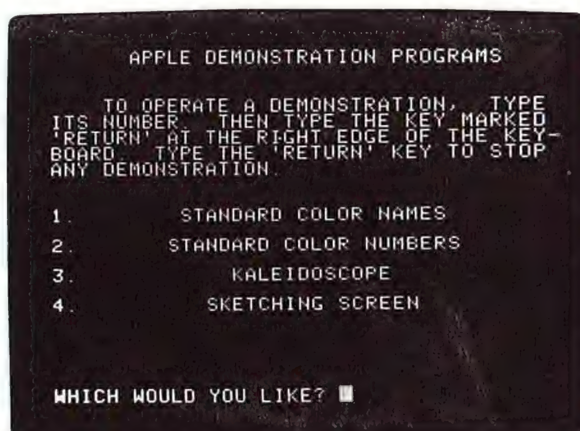
Sogleich erscheint eine Liste von Dateinamen auf dem Bildschirm.



Als erstes werden Sie das Programm mit dem Namen COLOR DEMOSOFT gebrauchen. Suchen Sie diesen Dateinamen in der Liste. Schreiben Sie dann

RUN COLOR DEMOSOFT

und drücken Sie anschließend die **RETURN** -Taste. Der Bildschirm sollte jetzt so aussehen :



Das Menü

Was Sie auf dem Bildschirm sehen, wird von Computerfritzen ein **Menü** genannt. Es funktioniert geradeso wie ein Menü in einem Restaurant. Sie wollen etwa einen strammen Max und sagen da „Herr Ober, Nummer 5 bitte". Suchen Sie sich also eine der möglichen Farbvorführungen aus, indem Sie die entsprechende Zahl schreiben. Selbstverständlich drücken Sie anschließend die **RETURN** -Taste. Und wenn Sie nach einer solchen Vorführung „zurück zum Menü wollen", drücken Sie nur die **RETURN** -Tasten.

Wie man den Computer anhält

Schreiben Sie





um den Computer anzuhalten. Damit erscheint wieder das Bereitschaftszeichen gefolgt vom blinkenden Cursor. Das Bereitschaftszeichen sagt Ihnen, daß Sie jetzt Befehle geben dürfen. Daher heißt es auch Bereitschaftszeichen, denn es zeigt die Bereitschaft des Computers an, nun Ihre Befehle auszuführen.

Sobald der Computer angehalten worden ist, können Sie ihn aufs neue mit

```
RUN
```

starten. (Natürlich kommt ein RETURN zum Abschluß, aber das braucht man Ihnen ja nun nicht mehr ständig zu sagen.)

Gebrauchen Sie





um den Computer zu stoppen,

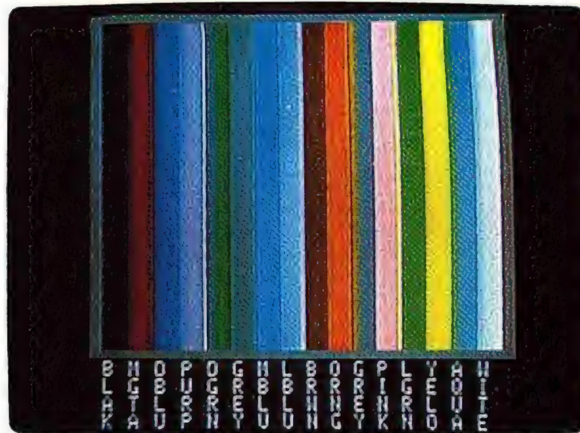
```
RUN
```

um ihn wieder zu starten. Versuchen Sie das ein paarmal.

Wie man die Bildschirmfarben einstellt

Wenn Sie das ,,Menü'' nicht mehr auf dem Bildschirm haben, starten Sie die Diskette erneut und lassen das Programm mit dem Namen COLOR DEMOSOFT laufen. Oder, wenn Sie ein Tonbandgerät statt der Diskette benutzen, benutzen Sie die übliche Prozedur zum Laden und Einlesen der Kassette ,,COLOR DEMOSOFT''. Eine der Menüeintragungen heißt STANDARD COLOR NAMES. Diese Vorführung werden wir dazu benutzen, die Farbeneinstellung des Fernsehers zu regulieren. Schreiben Sie also eine 1 und RETURN , um die besagte Vorführung auf den Bildschirm zu bringen. Sie erhalten daraufhin Streifen in verschiedenen Schattierungen oder sogar Farben. Unter jedem Streifen steht eine vierbuchstabige Abkürzung der Farbe. Es handelt sich dabei um die folgenden Farben, von links nach rechts :

- 0 Schwarz (BLACK)
- 1 Magenta - ein bläuliches Rot
- 2 Dunkelblau (DARK BLUE)
- 3 Lila (PURPLE)
- 4 Dunkelgrün (DARK GREEN)
- 5 Grau (GREY)
- 6 Mittleres Blau (MEDIUM BLUE)
- 7 Hellblau (LIGHT BLUE)
- 8 Braun (BROWN)
- 9 Orange (ORANGE)
- 10 Grau (GREY)
- 11 Rosa (PINK)
- 12 Grün (GREEN)
- 13 Gelb (YELLOW)
- 14 Aqua - halbwegs zwischen grün und blau
- 15 Weiß (WHITE)



Wenn Sie einen Schwarzweißfernseher oder einen Schwarzweißmonitor haben, stellen Sie Helligkeit und Kontrast nach Ihrem Geschmack ein. Natürlich stellen Sie den Fernseher wie gewöhnlich so ein, daß das Bild nicht nach oben oder unten wegwandert. Bei einem Farbfernseher müssen Sie etwas mehr einstellen.



Die Farben im obigen Bild sehen im PAL-System etwas anders aus, da dieses System nicht in den USA benutzt wird.
Vergessen Sie nicht, Farbempfinden ist subjektiv, und Sie können sich diese Farben ganz nach Geschmack einstellen. Mit den folgenden Befehlen können Sie die Farben so einstellen, wie es **uns** gefällt. Aber Sie sollten nach Ihrem Geschmack handeln, und ohnehin variieren die optimalen Einstellungen mit dem Licht im Zimmer.

Schalten Sie zunächst die automatische Farbregulierung ab. Das ist auf manchen Geräten ein mit „AUTO" markierter Schalter. Stellen Sie die Lautstärke leise ein, aber schalten Sie das Gerät nicht ab. Nun werden zwei Knöpfe wichtig: Helligkeit und Farbe. Stellen Sie die Helligkeit zunächst so dunkel ein, bis der Hintergrund fast völlig schwarz ist. Stellen Sie nun die Farbe auf etwa Mitte ein. Nun können sie die Helligkeit so einstellen, daß die Farben gut zu sehen sind, aber nicht an den Seiten der Streifen „auslaufen".

Wenn die Farben nun gut eingestellt sind, drücken Sie die **RETURN** -Taste, und das Menü ist wieder da. Versuchen Sie es nun mit DEMO 2, das Ihnen die Farben und ihre Kodierung angibt. Versuchen Sie auch die anderen Zahlenschlüssel des Menüs. Sie glauben ja nicht, was in so einem Fernseher alles steckt, bis Sie ihn an den Apple anschließen!

Das Spiel „Little Brick Out"

Lassen Sie das Programm LITTLE BRICK OUT laufen. Wenn Sie eine Diskette haben, sagen Sie einfach RUN LITTLE BRICK OUT. Beim Tonbandgerät müssen Sie die Kassette mit der Aufschrift LITTLE BRICK OUT einlesen. Wenn Sie nun das Programm laufen lassen, wird der Bildschirm wie auf dem linken Bild aussehen. Wenn Sie jetzt die Leertaste drücken, wird die Spielbeschreibung auf dem Bildschirm erscheinen.





Sie werden zunächst nach Ihrem Namen gefragt werden. Tippen Sie ihn ein und drücken Sie die **RETURN** -Taste. Wir schreiben als Beispiel

J. APPLESEED

Nun reagiert der Apple, indem er das Spielbrett zeichnet, einschließlich des Schlägers. Die Zahl gerade unter dieser Zeichnung ist die Punktzahl, die Sie bekommen können, wenn Sie die Rechtecke in den jeweiligen Spalten wegschlagen. Je tiefer Sie in die Wand dieser Rechtecke eindringen, desto mehr Punkte bekommen Sie. Sollte Ihr Name länger als 12 Buchstaben sein, so wird der Rest abgeschnitten, denn das Programm mag handlich kurze Namen lieber.

Wenn Sie zufällig **RESET** statt **RETURN** drücken, wird sich der Bildschirm aufhellen. Keine Unruhe! Schreiben Sie einfach

RUN

und es geht weiter. Beachten Sie, daß wir das abschließende **RETURN** schon mehrmals nicht erwähnt haben.

Üben Sie auch bewußt solche „Fehler", wie etwa die **RESET** -Taste zu drücken, so daß Ihnen vertraut wird, wie man diese Dinge rückgängig macht.

Mittlerweile teilt Ihnen das Programm mit

PUSH PADDLE BUTTON TO BEGIN GAME

(Drücken Sie den Knopf an der Spielkontrolle, um mit dem Spiel zu beginnen.) „Welche Spielkontrolle?" werden Sie sagen. Versuchen Sie beide! Sie sehen es dann schon, denn der Apple wird Ihnen mitteilen, ob Sie die falsche erwischen.

KAPITEL 2
Applesoft für Anfänger

Applesoft für Anfänger

Wenn Sie im Applesoft sind, dann wird auf einen Druck auf die RETURN-Taste hin an der linken Bildschirmkante eine eckige Klammer (]) als Bereitschaftszeichen, gefolgt von dem blinkenden Cursor, erscheinen. Sonst müssen Sie Applesoft laden. Falls Sie ein Floppy Laufwerk haben, bringen Sie DOS zum Laufen.

Ein erster Blick auf den Schreibbefehl

Jetzt, wo Sie das Applesoft Bereitschaftszeichen (]) und den blinkenden Cursor auf dem Schirm haben, können Sie Applesoft benutzen. Schreiben Sie :

PRINT "HELLO"

und der Computer wird das Wort

HELLO

auf der nächsten Zeile schreiben. Wenn er es nicht tut, müssen Sie prüfen, ob Sie nicht vergessen haben, die RETURN-Taste zu drücken. Wenn Sie das Wort "PRINT" falsch schreiben, werden Sie die Nachricht

? SYNTAX ERROR

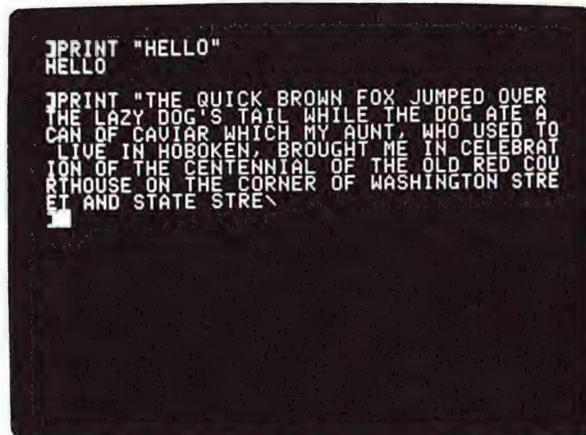
bekommen. Wenn Sie entweder die ersten oder gar beide Anführungsstriche (wir werden sie auch Apostrophe nennen) vergessen, wird der Computer eine Null schreiben. Sie können am schrägen Strich erkennen, daß es sich um die Null handelt.

Ø

Sie können die abschließende Apostrophe (Anführungszeichen oben) weglassen, wenn es das letzte Zeichen vor dem RETURN ist. Das Wort "HELLO" wird mit oder ohne die abschließende Apostrophe geschrieben werden. Es wäre jedoch besser, dennoch die abschließende Apostrophe zu schreiben. Diese Angewohnheit wird später wichtig werden. In diesem Handbuch werden wir also annehmen, daß Sie die abschließende Apostrophe (Anführungszeichen oben) immer setzen.

PRINT "HELLO"

ist ein Befehl an den Computer, die Zeichen zwischen den Apostrophen auf dem Bildschirm zu schreiben; in diesem Falle also einen Gruß. Sie können den PRINT Befehl dazu benutzen, um den Computer jede Nachricht schreiben zu lassen.
Wenn Sie aber mehr als 24Ø Zeichen schreiben lassen, wird der Computer piepen und das Zeichen „\" schreiben. Danach dürfen Sie wieder von vorne anfangen.



Versuchen Sie jetzt das Folgende

PRINT "150"

Gehorsam schreibt der Computer die Zahl 150 auf die nächste Zeile. Schreiben Sie

PRINT 150

und der Computer schreibt wieder die Zahl, ohne sich zu beklagen oder eine Fehlermeldung, daß Apostrophe fehlen, zu geben. Es ist sogar so, daß Apple Sie mit PRINT jede Zahl schreiben lassen wird, ohne daß Sie Apostrophe benutzen zu brauchen.

Ohne weiteres Studium können Sie jetzt den Apple als einfachen Tischrechner benutzen.

Versuchen Sie auf Ihrem Apple

PRINT 3+4

Die Antwort 7 erscheint auf der nächsten Zeile. Der Apple kann sechs verschiedene elementare arithmetische Operationen ausführen :

1. ADDITION. Man benutzt das übliche Plussymbol (+).

2. SUBTRAKTION. Benutzen Sie das Minussymbol (—).

3. MULTIPLIKATION. Viele Leute benutzen ein „x" als Multiplikationssymbol. Aber man könnte das mit dem Buchstaben „x" verwechseln.

Einige Leute benutzen einen Hochpunkt (·). Aber es könnte mit einem Punkt verwechselt werden. Der Apple benutzt einen Stern (*). Um herauszufinden, was 7*8 ist (falls Sie sich an die Antwort nicht erinnern können), schreiben Sie bloß

PRINT 7*8

und Ihr Gedächtnis wird aufgefrischt.

4. DIVISION. Wir benutzen hierfür einen schrägen Strich (/). Um 63 durch 7 zu teilen, schreiben Sie

PRINT 63 / 7

und die richtige Antwort wird erscheinen.
Versuchen Sie 3 durch 2 zu teilen. Die Antwort ist eineinhalb.
Der Apple gibt Ihnen die Antwort mit einem Dezimalpunkt statt eines Kommas : 1.5.

Wir sollten betonen, daß man mehr als eine arithmetische Operation in demselben Befehl ausführen kann. Zum Beispiel kann man

PRINT 3 + 5 + 9 + 4

schreiben. Die genauen Regeln, wie solche Ausdrücke zu schreiben sind, werden wir erst später erklären, aber Sie können jetzt schon damit experimentieren.

6. POTENZIEREN. Es ist oft nützlich eine Zahl mit sich selbst mehrmals zu multiplizieren. Statt zu schreiben

PRINT 4*4*4*4*4*4

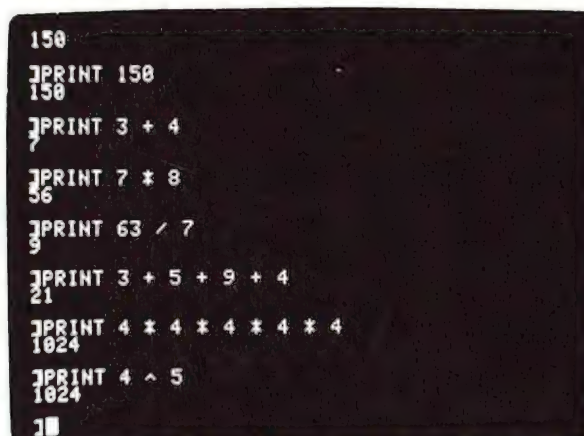
können Sie abgekürzt

PRINT 4^5

(„4 hoch 5") schreiben. Wir benutzen den nach oben zeigenden Pfeil :



Potenzieren ist nichts Besonderes. Es ist nur eine Abkürzung für wiederholtes Multiplizieren. In der mathematischen Schreibweise, die nicht speziell für den Computer zugeschnitten ist, würde man 4^5 schreiben.



Applesofts Zahlenformat

Schreiben Sie

PRINT 45.340

Ihr Computer antwortet mit

45.34

und unterschlägt die letzte Null. Der Apple schreibt beim PRINT weder davorgesetzte, noch nachfolgende Nullen. Sehr kleine Zahlen, ungefähr zwischen 0.000000000000000000000000000000000000003 und -0.000000000000000000000000000000000000003, werden vom Apple in null verwandelt. (Hoffentlich sind die Nullen richtig! Es ist aber leichter, solche Zahlen folgendermaßen zu schreiben : 3 * 10 ^ -39 und -3 * 10 ^ -39. Glauben Sie das aber nicht, ohne es selbst zu probieren!
Schreiben Sie nun

PRINT 985788.6898

Eine Überraschung! Die zwei letzten Dezimalstellen sind verloren gegangen, und der Computer hat die am besten angenäherte Zahl gewählt, die er sich denken konnte. So etwas nennt sich **runden.**
Versuchen Sie es nun mal mit

PRINT 788.6898

Jetzt hat die Maschine nicht gerundet, sondern die Zahl so wie sie war geschrieben. Verrückt, werden Sie sagen! Ah, aber dieser Wahnsinn hat Methodik! Zahlen werden nämlich nur dann gerundet, wenn sie mehr als 9 Ziffern haben. Jede Zahl mit weniger als zehn Ziffern bleibt ungerundet und exakt. Ja, der Computer bemüht sich schon, er kann aber eben nur mit neun Ziffern rechnen.

Wenn Sie einen PRINT Befehl mit großen Zahlen geben, etwa

PRINT 1234567890

dann antwortet der Apple darauf mit

1. 23456789E+09

Die Zahlen 1234567890 und 1.23456789E+09 sind gleich. Wirklich! Die vom Computer geschriebene Zahl ist in einem besonderen Format, das oft von Ingenieuren benutzt wird. Wenn Sie dieses Format noch nicht kennen und mehr darüber wissen möchten, finden Sie darüber im Applesoft Programmierhandbuch weitere Informationen.

Machen Sie andere Experimente. Finden Sie heraus, ab welcher Zahlengröße der Apple Ihre Zahlen in das besondere Format überträgt. Und falls Sie das Format zu kompliziert finden, brauchen Sie sich nicht zu sorgen, denn wahrscheinlich werden Sie so große Zahlen ohnehin für lange Zeit nicht gebrauchen. Beachten Sie außerdem, daß in Apostrophe eingeschlossene Zahlen genau so gedruckt werden, wie sie geschrieben wurden. Allerdings können Sie mit solchen Zahlen nicht rechnen. Das Applesoft II BASIC Programmierhandbuch wird Ihnen alles Wissenswerte über Zahlenformate mitteilen.

Mehr von RETURN

Bisher haben Sie die RETURN- Taste wie ein Automat nach jeder Zeile gedrückt. Vielleicht sollten wir Ihnen nun erklären, wieso diese Taste so oft gebraucht wird. Ganz einfach : Ohne ein RETURN weiß der Computer nicht, ob Sie mit Ihrem Befehl schon fertig sind. Sie könnten zum Beispiel anfangen mit

PRINT 4+ 5

Wenn nun der Computer gleich eilfertig eine 9 schriebe, könnte es Sie verärgern. Denn Sie hätten durchaus planen können,

PRINT 4 + 5 + 346

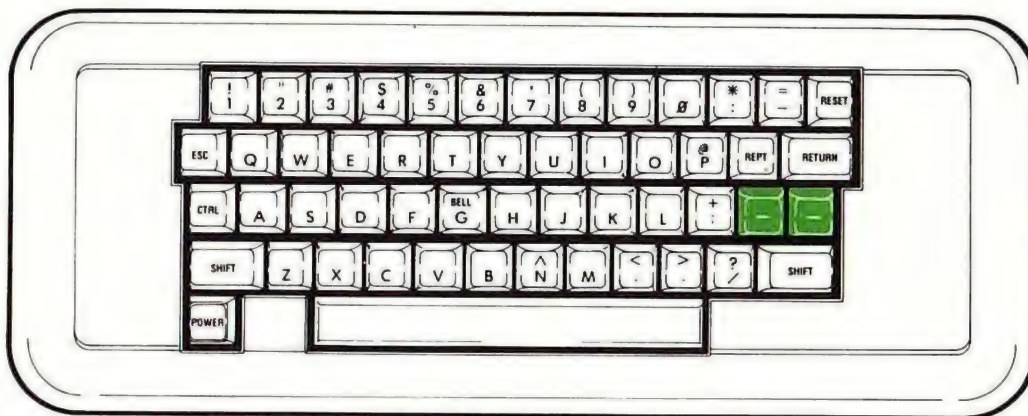
zu schreiben, was ja wohl eine andere Antwort ergäbe. Da der Computer nicht hellsehen kann, wann Sie nun wirklich fertig sind, müssen Sie es schon dem Computer selber sagen, und das tun Sie durch einen Druck auf die RETURN-Taste. Da Sie die Taste nach jedem Befehl drücken müssen, haben wir schon längst aufgehört, diesen Sachverhalt zum guten Abschluß zu erwähnen.
Wenn Sie schön fleißig alle bisherigen Beispiele mitgemacht haben, ist es Ihnen auch schon in Fleisch und Blut übergegangen.

Wirklich, es wäre uns lieb, wenn Sie alle Beispiele mitmachten. Programmieren zu lernen ist ja wie Fahrradfahren oder Klavierspielen. Sie können da ganze Bibliotheken mit Theorie lesen, aber ohne es zu üben, wird es Ihnen nicht von der Hand gehen. Wie beim Fahrradfahren lernen Sie programmieren durch (manchesmal schmerzliche) Erfahrung.

Wenn Sie nun das Fahrradfahren erlernt haben, können Sie bald überall hin. Dasselbe gilt für die Programmierung. Sie können dieses Buch bloß lesen und vielleicht denken, Sie verstehen alles. Deshalb können Sie trotzdem noch lange nicht programmieren.
Sie müssen schon die Beispiele mitmachen, sonst wird daraus nichts, das ist die nackte Wahrheit.

Einfache Aufbereitungsbefehle oder was man vor RETURN machen kann

Niemand ist perfekt im Tippen. Wir machen sogar Feeler (Hoppla!) Der Apple hat verschiedene Dinge, die Ihnen helfen können, Fehler zu berichtigen. Er erspart Ihnen damit, die gesamte Zeile wegen eines blöden Fehlers neu zu schreiben. Dabei handelt es sich um die Tasten, die mit einem Links- und einem Rechtspfeil markiert sind.



Die ← Taste funktioniert fast so wie die entsprechend markierte Taste bei Schreibmaschinen, und sie wird auch BACKSPACE Taste genannt. Einige Experimente werden das verdeutlichen. Schreiben Sie so wie es da steht

PRINT COMPUTER"

Drücken Sie, wie üblich, die RETURN-Taste. Der Computer antwortet

0

wegen der fehlenden Apostrophe. Wenn wir

PRINT "COMPUTER"

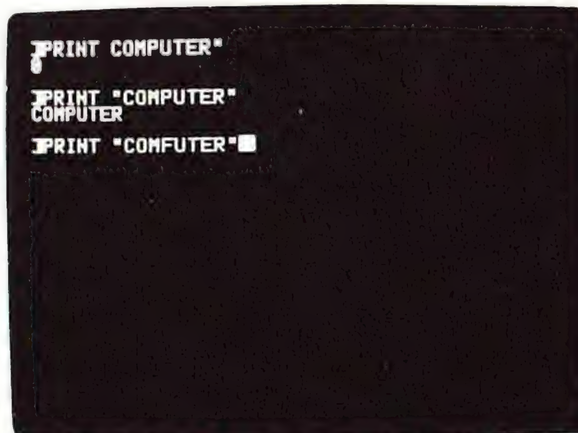
geschrieben hätten, so wäre als Antwort

COMPUTER

erschienen. Glauben Sie uns das alles nicht, ohne es selbst zu probieren. Schreiben Sie nun den fehlerhaften Befehl - **ohne RETURN zu drücken**

PRINT "COMFUTER"

Da Sie noch nicht RETURN gedrückt haben, ist auch noch nichts passiert. Wie Sie auf dem Bild hier sehen, ist der Cursor immer noch rechts neben der abschließenden Apostrophe.



Um

COMFUTER

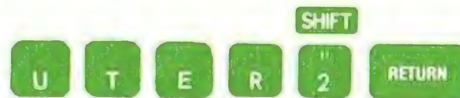
in

COMPUTER

umzuändern, benutzen wir die ← Taste. Beachten Sie, daß bei jedem Druck dieser Taste der blinkende Cursor sich um eine Stelle nach rechts bewegt. Wir haben also den Cursor „zurückbewegt" (backspace im Englischen). Bewegen Sie also den Cursor zurück zum F, und schreiben Sie statt dessen ein P. Sie sehen, das F wurde durch das P ersetzt. Drücken Sie nun RETURN. Wie Sie sehen, bekommen Sie

COMP

und das von COMPUTER? Das liegt daran, daß Sie über UTER mit dem Cursor nach links gingen, und die Zeichen, über die Sie zurücksetzen, werden dann nicht auf RETURN hin an die Maschine geschickt. Natürlich könnten Sie das korrigieren, indem Sie anschließend



schreiben. Versuchen Sie es, es funktioniert!

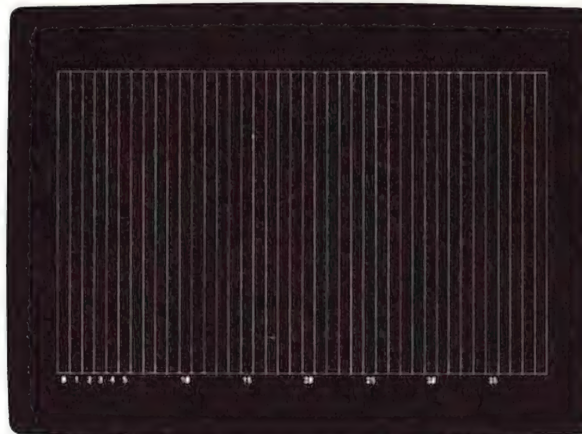
Aber es geht auch einfacher. Wenn Sie die → Taste drücken, bewegt sich der Cursor nach rechts. Bei jeder Rechtsbewegung ist es so, als ob das übersprungene Zeichen wieder neu eingetippt würde. Daher nennen wir die → Taste auch RETYPE Taste, denn RETYPE bedeutet „neu schreiben". Schreiben Sie nochmal

PRINT "COMFUTER"

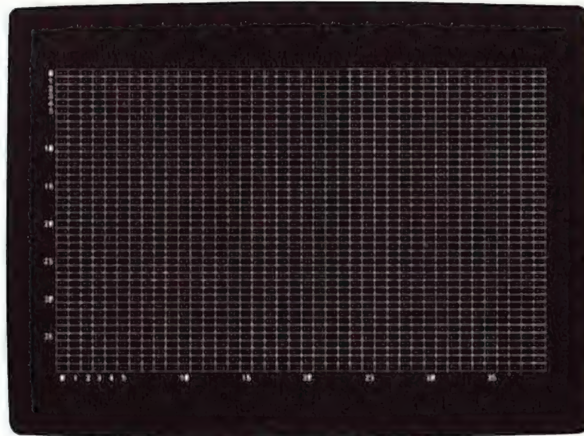
Bewegen Sie dann den Cursor nach links über das F, ändern Sie es in ein P ab, und bewegen Sie nun den Cursor wieder fünf Stellen mit der RETYPE Taste nach rechts. Drücken Sie nun RETURN. Hat alles geklappt? Die BACKSPACE- und die RETYPE-Taste werden Ihnen noch viel Zeit sparen! Üben Sie die Tasten mit gewollten Fehlern, sodaß Sie mit ihnen vertraut werden.

Wie man den Bildschirm farbig macht

Um farbige Grafiken auf dem Bildschirm zu malen, brauchen wir eine Sprache, in der man so etwas beschreiben kann. Wir haben insgesamt 16 Farben. Um zu beschreiben, wo wir auf dem Bildschirm nun eine gewisse Farbe haben möchten, teilen wir zunächst den Bildschirm in 40 Spalten ein. Wir zählen die Spalten von 0 bis 39 von links nach rechts. Sie fragen sich vielleicht, warum wir nicht mit 1 statt 0 bei der Numerierung anfangen?
Nun, während Ihre Programmierkunst im Lauf der Zeit wächst, werden Sie sehen, daß es so einfacher ist, obwohl man das auf den ersten Blick veilleicht nicht gleich sieht.



Nun teilen wir den Bildschirm in 40 Zeilen auf, wieder von 0 bis 39 numeriert, und zwar von oben nach unten aufsteigend. 39 ist also die unterste Zeile. Dadurch erhalten wir 40 mal 40 Rechtecke.
Die Experten sagen jetzt: Aha! Einfach ein kartesisches Koordinatensystem. Nun, wenn Ihnen solche Fachausdrücke unangenehm sind, können wir ja auch von Zeilen und Spalten reden.



Um bunte Bilder auf dem Schirm zu malen, schreiben Sie zunächst

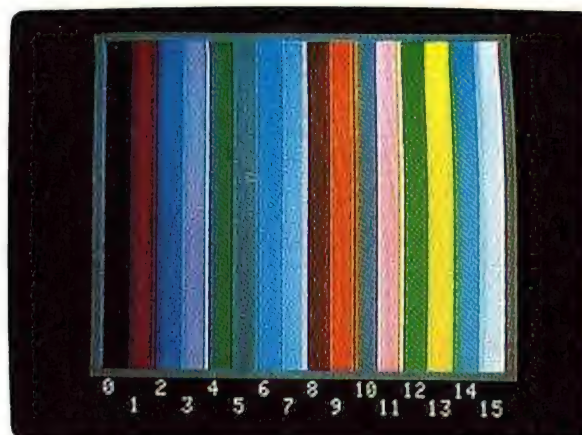
GR

Natürlich haben Sie an **RETURN** gedacht. Wenn Sie diesen Befehl geben, wird der Bildschirm gelöscht. Lediglich vier Zeilen Text bleiben unten stehen. „GR" kürzt Grafik ab. Wenn Sie alles wieder so wie vorher haben wollen, schreiben Sie

TEXT

Wenn Sie diesen Befehl geben, erscheinen plötzlich viele „Klammeraffen" (@) auf dem Bildschirm. Das ist ganz normal. Probieren Sie den TEXT-Befehl, kehren Sie danach wieder zu den Grafiken mit dem GR-Befehl zurück.

Bevor Sie nun einen farbigen Punkt auf den Bildschirm setzen, müssen Sie zuallererst dem Computer sagen, welche Farbe Sie möchten.
Sie haben die Wahl zwischen 16 Farben, das sahen Sie ja schon vorhin mit dem Programm COLOR DEMOSOFT 2. Die Farben werden von Ø bis 15 numeriert.



Nehmen wir an, Sie wollten einen grünen Klecks irgendwohin zeichnen.
Dazu geben Sie zuerst den GR Befehl, dann schreiben Sie

COLOR = 12

Von nun an wird jeder Punkt (also die Rechtecke von vorhin), den Sie zeichnen, grün sein. Das wird so bleiben, bis Sie dem Computer eine andere Farbe vorschreiben. Natürlich ist der Bereich unten auf dem Bildschirm ausgeschlossen und für Ihre Befehle reserviert. Um nun einen farbigen Punkt in die linke obere Ecke zu zeichnen, also in die Spalte Ø und Zeile Ø, sagen Sie

```
PLOT Ø,Ø
```

Um einen Punkt gleicher Farbe in die rechte obere Ecke zu zeichnen, also in Spalte 39 und Zeile Ø, schreiben Sie

```
PLOT 39,Ø
```

Beachten Sie, daß wir die Spaltennummer immer zuerst nennen.
Lassen Sie uns nun einen orangenen Punkt nach unten links setzen. Zuerst verändern wir die Farbe -- übrigens, Sie sollten wirklich diese Experimente mitmachen! Spucken Sie sich also in die Hände und schreiben Sie

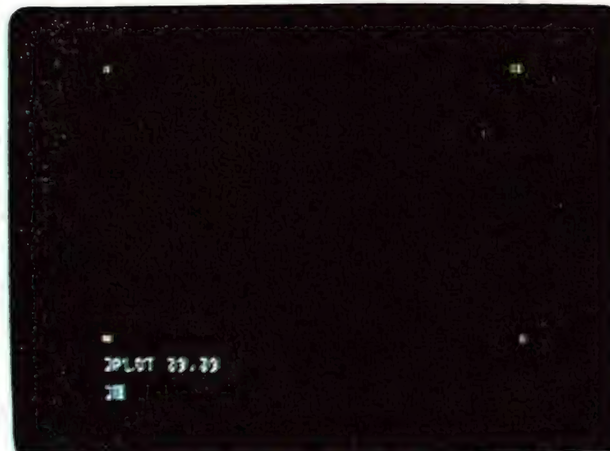
```
COLOR= 9
```

Die oben gezeichneten Punkte verändern sich überhaupt nicht, sogar wenn Sie sich an **RETURN** erinnerten. Aber der Computer hat sich die neue Farbe gemerkt, und was Sie nun zeichnen, wird orange und nicht grün sein. Nun haben wir die Farbe gewählt, und wird können jetzt unseren Punkt in die untere linke Ecke, also in Spalte Ø und Zeile 39, Setzen :

```
PLOT Ø,39
```

War das richtig? Sie haben auch **RETURN** nicht vergessen? Mögen Sie diese Farbe?

Wir werden nun einen magentafarbenen Punkt in die untere rechte Ecke einzeichnen. Knobeln Sie doch mal selbst heraus, wie man es hinschreiben muß.



Grafik Fehlermeldungen

Zwei Fehlermeldungen können Ihnen beim Grafikzeichnen leicht unterlaufen. Wenn Sie sich verschreiben und etwa

```
PLAT
```

oder

PLOP

statt

PLOT

schreiben, bekommen Sie die Fehlermeldung

? SYNTAX ERROR

Das wissen Sie ja schon. Sie bekommen eine neue Fehlermeldung zu Gesicht, wenn Sie mit Koordinaten, die nicht zwischen Ø und 39 liegen, zu zeichnen versuchen. Schreiben Sie

PLOT 13,85

Sie werden

? ILLEGAL QUANTITY ERROR

lesen. Das heißt, Sie haben versucht, einen Punkt außerhalb des Bildschirmbereichs zu zeichnen. Die größte verwendbare Zahl für die erste Koordinate ist 39, 47 für die zweite. Wenn Sie allerdings Werte über 39 für die zweite Koordinate verwenden, wie etwa in

PLOT 2Ø,45

so bekommen Sie lediglich eigenartige Zeichen im Textbereich unten im Bildschirm zu sehen.

Auch wenn Sie versuchen, negative Koordinatenwerte zu benutzen, wird man es Ihnen mit

? ILLEGAL QUANTITY ERROR

lohnen.

Wie man Linien zeichnet

Nehmen wir an, Sie hätten gern eine waagerechte Linie in der Zeile 14 von Spalte 5 bis Spalte 9 gezeichnet, und zwar hellblau. Natürlich können Sie nun schreiben

```
COLOR = 7
PLOT 5,14
PLOT 6,14
PLOT 7,14
PLOT 8,14
PLOT 9,14
```

Beachten Sie, daß es da keinen Zwischenraum zwischen anstoßenden Rechtecken gibt. Sie bilden eine durchgehende Linie. Aber es gibt auch einen leichteren Weg, horizontale Linien zu zeichnen.
Das wäre ja noch schöner, wenn es nicht so wäre. Nehmen wir an, Sie möchten eine dunkelgrüne horizontale Linie in der Mitte quer über dem Bildschirm gezeichnet haben. Der lange Weg dazu benötigt etwa vierzig Befehle :

```
COLOR = 4
PLOT 0,20
PLOT 1,20
PLOT 2,20
```

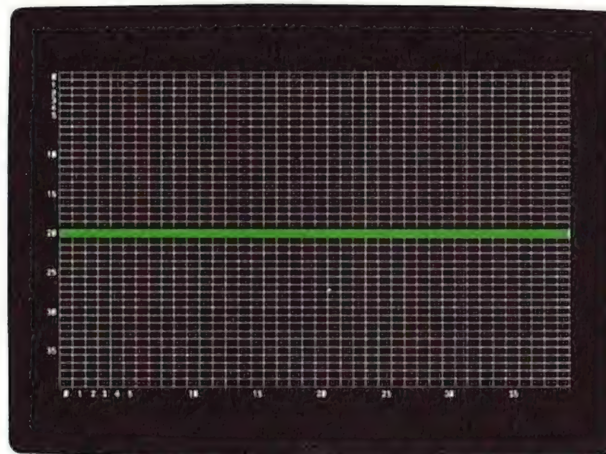
und so weiter, bis endlich zu

```
PLOT 39,20
```

Einfacher wäre das Folgende :

```
COLOR = 4
HLIN 0,39 AT 20
```

Drücken Sie **RETURN**, bitteschön, da haben Sie es schon : Sofort eine waagerechte Linie in der Zeile 20 quer über den Bildschirm von Spalte 0 bis 39.

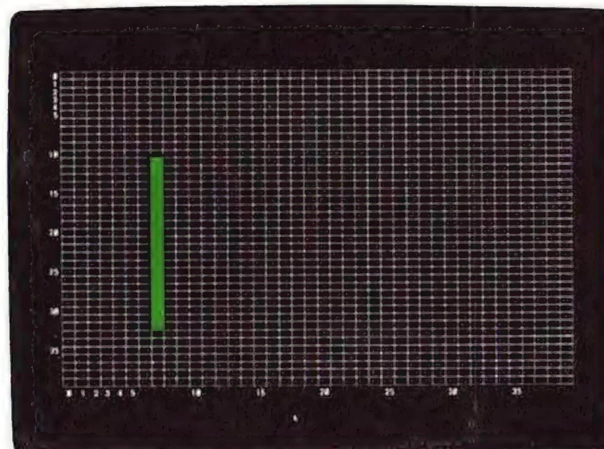


Versuchen Sie nun, eine lila Linie in Zeile 18 von Spalte 19 bis 28 zu zeichnen. Versuchen Sie andere Linien. Nach etwa 6 Linien werden Sie es heraushaben.

Beachten Sie, wenn man einen farbigen Punkt oder Linie auf eine Stelle setzt, die schon eine andere Farbe hat, übermalt die neue Farbe die alte. Um den Bildschirm völlig und auf einmal zu löschen, geben Sie erneut den GR Befehl.

Es gibt einen entsprechenden Befehl, senkrechte Linien zu zeichnen. Um also eine vertikale Linie in der Spalte 7 von Linie 12 nach Linie 33 zu zeichnen, schreiben wir :

```
COLOR= 9
VLIN 12,33 AT 7
```



Üben Sie, vertikale Linien in anderen Spalten und mit verschiedenen Anfangs- und Endpunkten zu zeichnen. Sie können das Erlernte unter Beweis stellen, indem Sie mit nur fünf Befehlen den Bildschirm rot einrahmen. Malen Sie danach ein grünes Kreuz.
Probieren Sie, Linien mit COLOR = Ø zu zeichnen. Spielen Sie mit PLOT, HLIN und VLIN herum. Dieses Buch wird Ihnen nichts nützen, wenn Sie nicht selbst experimentieren.

Die Spielkontrollen

Nehmen Sie die Spielkontrolle her, mit der Sie LITTLE BRICK OUT gespielt haben. Schreiben Sie mit der einen Hand

```
PRINT PDL (Ø)
```

-- eine Zahl sollte erscheinen. Bewegen Sie den Drehknopf mit der anderen Hand. Schreiben Sie wieder

```
PRINT PDL (Ø)
```

Experimentieren Sie mit den Knöpfen, und schreiben Sie abwechselnd

```
PRINT PDL (Ø)
```

Wenn die geschriebenen Zahlen gleich bleiben, drehen Sie an der falschen Spielkontrolle. Welches ist die niedrigste, welches die höchste Zahl, die Sie geschrieben sehen? Um wie wenig kann sich die Zahl beim Drehen des Drehknopfes verändern?

Sie können die Stellung der anderen Spielkontrolle herausfinden, indem Sie PRINT PDL (1) schreiben. Die Abkürzung PDL ist für Paddle, das sowohl ein Paddel als auch ein Tischtennisschläger sein kann.
Wir nennen es so, veil in vielen Spielen damit ein Schläger bewegt wird. Es gibt auch noch viele andere Verwendungszwecke für die Spielkontrollen.

PDL ist eine **Funktion.** Eine Applesoft-Funktion ist etwas, das ein oder mehrere Zahlen als Argument nimmt, mit ihnen irgendwie rechnet, und dann einen einzigen Wert als Resultat abliefert.
Argumente werden immer in Klammern gesetzt, dem Funktionsnamen folgend. PDL ist also eine Funktion mit einem Argument (einstellig). Die Zahl, die das Ergebnis ist, wird, wie man sagt, „zurückgebracht" oder „geliefert".

Speicher und weitere Rechenfähigkeiten

Bei vielen Taschenrechnern ist es möglich, eine Zahl für spätere Weiterverwendung zunächst abzuspeichern. Dazu weist man die Zahl einem bestimmten Ort im Rechner zu. Nennen wir so einen Platz zunächst einmal ein Speicherplatz. Häufig drückt man dazu bei Taschenrechnern eine mit „M" bezeichnete Taste („M" für „Memory" - also Gedächtnis). Auf dem Apple können Sie das auch. Um z.B. den Wert 77 zu speichern, schreiben Sie

```
M = 77
```

Dieser Wert 77 wird nun nicht etwa geschrieben, sondern in einem Speicherplatz aufbewahrt, der M benannt wird. Wenn Sie nun schreiben

PRINT M

Siehe da, der Computer druckt den Wert von M. Geben Sie diese beiden Befehle ein.

Schreiben Sie jetzt

M = 324

Schreiben Sie mit PRINT nochmals den Wert von M. Der Wert ist jetzt 324, nicht wahr? Und was passierte mit der 77? Nun, die ist für immer weg. Ein Speicherplatz kann also immer nur einen Wert auf einmal haben, und wenn Sie einen neuen Wert abspeichern, geht der alte verloren.

Schreiben Sie

PRINT "M"

Was passiert? Zwischen

M

und

"M"

besteht also ein großer Unterschied!

Dieser Unterschied ist fast so wie der Unterschied zwischen :

Eine Maus hat vier Pfoten.

und

"Maus" hat vier Buchstaben.

Im ersten Fall reden wie von einem kleinen Tier mit einem langen Schwanz, im anderen Fall reden wir vom Wort selbst. So werden die Apostrophe im Computer-Chinesisch benutzt. Wenn wir

PRINT "M"

schreiben, meinen wir, daß der Buchstabe selbst gedruckt werden soll. Und mit

PRINT M

meinen wir, daß die Bedeutung von M geschrieben werden soll.
Sie würden auch nicht den Namen Ihrer Geliebten mit Ihrer Geliebten selbst verwechseln.

Man kann das Ergebnis einer Rechnung abspeichern.
Zum Beispiel

M = 3 + 4

mit dem PRINT M Befehl können Sie sehen, was geschieht.

Sie dürfen auch den Wert von M in einer weiteren Rechnung benutzen. Versuchen Sie zum Beispiel

PRINT M + 2

auf Ihrem Apple. Ist die Antwort wie erwartet? Versuchen Sie andere Rechnungen mit M.

Ein einfacher Taschenrechner hat bloß einen Speicher. Computer haben dagegen hunderte, und Applesoft hat 936. Gewöhnlich werden diese Variablen genannt. Der Name ist etwas irreführend, da sich Variablen eines Computers anders verhalten als Variablen in der Mathematik. Viel einfacher nämlich.
Jede Variable ist eben nur ein Fach, in dem man einen Wert abspeichern kann. Nun, Variable hat sich als Name eingebürgert.
Wir werden unsere Speicherplätze also ebenso nennen - Sie brauchen bloß Ihre mathematischen Kenntnisse zu vergessen. Alle Variablen haben beim Apple den Wert 0, bis Sie ihnen einen neuen Wert zuweisen.

Eine Variable darf mit fast jedem Namen benannt werden, den Sie sich denken können, bloß muß er mit einem Buchstaben anfangen. Zum Beispiel :

```
SUM = 56 + 34 + 1523 + 8
GAMEPOINTS = 45
PLAYER2 = 9
```

Manche Namen sind verboten, weil in ihnen ein Teilwort vorkommt, das für den Apple eine besondere Bedeutung hat. Die Worte mit besonderer Bedeutung werden reservierte Worte genannt. Eines dieser reservierten Worte ist COLOR. Daher darf COLOR nicht in einem Variablennamen vorkommen. Schreiben Sie zum Beispiel

```
THIS COLOR = 6
```

oder

```
COLORFUL = 9
```

da bekommen Sie nur Fehlermeldungen. Jedesmal, wenn ein Variablenname die Meldung ?SYNTAX ERROR verursacht, haben Sie (un)wissentlich ein reserviertes Wort in diesem Namen. Kein Grund zur Beunruhigung, Sie suchen sich einfach einen anderen Namen aus.

Im Anhang B haben wir Ihnen eine Liste aller reservierten Worte im Applesoft zusammengestellt.

Es ist eine gute Idee, Variablennamen derart zu wählen, daß sie sich auf den Verwendungszweck der Variablen beziehen. Dann haben Sie es einfacher, sich an die Namen zu erinnern.

Versuchen Sie es mit

```
VOGEL = 11
```

und danach

```
PRINT VOGEL
```

Ging alles wie erwartet? Schreiben Sie nun

```
PRINT VOLL
```

Was wird geschrieben? Versuchen Sie es nun mit

```
PRINT VORHER
```

und

```
PRINT VORBEI
```

Wenn Sie sich diese Namen genauer ansehen, bemerken Sie, daß alle mit denselben zwei ersten Buchstaben beginnen, nämlich mit "VO".
Applesoft benutzt nur die ersten zwei Zeichen eines Namens, um ihn von anderen Namen zu unterscheiden. Daher beziehen sich die Namen

```
VOGEL
```

und

```
VOLL
```

auf die gleiche Variable wie

```
VORHER

VORBEI
```

usw.

Hier noch ein nützlicher Hinweis. Nehmen wir an, Sie haben eine Variable "PRICE", und Sie möchten den Wert der Variable um 5 erhöhen. Sie können dafür zunächst den Wert mit PRINT sich geben lassen, danach 5 addieren lassen und schließlich das Ganze PRICE zuweisen. Zum Beispiel

```
PRICE = 28
PRINT PRICE
PRINT 28 + 5
PRICE = 33
```

Aber wieviel einfacher ist doch

```
PRICE = 28
PRICE = PRICE + 5
```

Versuchen Sie die folgenden Befehle in der Reihenfolge, in der wir sie schrieben :

```
PRICE = 2
PRINT PRICE
PRICE = PRICE + 3
PRINT PRICE
PRICE = PRICE * 6
PRINT PRICE
PRICE = PRICE / 10
PRINT PRICE
```

Zum Schluß müßte nun eine 3 dastehen, nicht wahr? Hatten Sie das auch erwartet? Versuchen Sie nun mal :

```
APPLES = 55
BANANAS = 11
QUOTIENT = APPLES / BANANAS
PRINT QUOTIENT
```

Denken Sie zuerst nach, was dabei herauskommen müßte. Prüfen Sie daraufhin, ob Sie es richtig errechnet haben. Falls Ihr Ergebnis falsch ist, müssen Sie sich überlegen, warum. Versuchen Sie schließlich noch

```
HELLO = 128
PRINT "HELLO"
HELLO = HELLO / 2
PRINT "HELLO"
HELLO = HELLO / 2
PRINT HELLO
```

Was hatten Sie erwartet? Welches Ergebnis erhielten Sie?

Vorrangigkeit oder wer ist zuerst dran?

Bei manchen formellen Tischgesellschaften wurden früher die Leute in einer bestimmten Reihenfolge bedient: Zuerst der Ehrengast, dann die Frauen (je nach ihrem gesellschaftlichen Rang) und dann die Männer (ebenfalls ihrem Rang nach). Der Gastgeber meistens ganz zuletzt. Ganz egal, wo wer saß, der Ober ging nach dieser Regel von Platz zu Platz. Wir könnten das Vorrangigkeit unter den Gästen nennen. In einer einfachen Rechnung, wie etwa

```
PRINT 4 + 8 / 2
```

können Sie nicht sicher sein, ob 6 oder 8 herauskommen soll, bis Sie wissen, mit welcher Vorrangigkeit die Operationen ausgeführt werden sollen. Wenn zuerst addiert wird, kommt 12 durch 2, also 6 heraus. Das wäre eine Möglichkeit. Wenn aber zuerst geteilt wird, so hätten wir 4 plus (8 / 2), also 4 plus 4, und somit wäre 8 auch eine mögliche Antwort. Der Apple wird Ihnen bei diesem Beispiel 8 errechnen. Wir haben Ihnen die Reihenfolge zusammengestellt, in der sich der Apple aussucht, was als nächstes berechnet werden soll:

1. Wenn das Minuszeichen benutzt wird, um anzuzeigen, daß die Zahl negativ sein soll, zum Beispiel in

```
- 3 + 2
```

dann wird der Apple zuerst dieses Minuszeichen anwenden, um die Zahl oder den Variablenwert zu negieren. Daher ist −3 + 2 gleich −1. Wenn wir zuerst addiert hätten, wäre −5 dabei herausgekommen.
Tut es aber nicht. Ein weiteres Beispiel:

```
BRIAN = 6
PRINT -BRIAN + 10
```

Dabei sollte 4 herauskommen. (Beachten Sie aber, in dem Ausdruck 5-3 bedeutet das Minuszeichen Subtraktion, nicht Negation einer Zahl).

2. Nachdem die Negation ausgeführt worden ist, kommt als nächste Operation die Potenzierung an die Reihe. Der Ausdruck

```
4 + 3 ^ 2
```

wird berechnet, indem die 3 zuerst quadriert wird (3 mal 3 ist 9), und dann wird 4 dazugezählt, so daß alles in allem 13 herauskommt. Wenn mehrere Potenzierungen durchgeführt werden sollen, so wird das von links nach rechts gemacht. Daher wird

```
2 ^ 3 ^ 2
```

berechnet, indem zunächst 2 hoch 3 ausgerechnet wird (2 mal 2 mal 2 ist nun 8), dann wird das Zwischenergebnis 8 danach quadriert, so daß 64 als Ergebnis ausgerechnet wird.

3. Nachdem die Potenzierungen ausgeführt worden ist, wird als nächstes jede Multiplikation und Division ausgeführt, von links nach rechts. Arithmetische Operationen gleicher Vorrangigkeit werden übrigens immer von links nach rechts berechnet. Multiplikation und Division (* und /) haben gleiche Vorrangigkeit.

4. Schließlich werden alle Additionen und Subtraktionen durchgeführt, wieder von links nach rechts. Addition (+) und Subtraktion (−) haben gleiche Vorrangigkeit.

Fassen wir nochmals die Vorrangigkeit der Operationen zusammen, die der Apple beachtet :

Zuerst : − (benutzt zum negieren von Werten)
Zweitens : ^ (Potenzierung, von links nach rechts)
Drittens : * / (Multiplikation und Division, von links nach rechts)
Viertens : + − (Addition und Subtraktion, von links nach rechts)

Wir haben auf der nächsten Seite einige arithmetische Ausdrücke zusammengestellt, die Sie auswerten lassen können. Jedem sollten Sie zuerst einmal im Kopf (oder mit Bleistift und Papier (berechnen, und ihn dann vom Apple berechnen lassen. Wenn Sie etwas anderes berechnet haben, müßten Sie sich überlegen, weshalb das so ist. Wir geben hier nur den Ausdruck selbst. Um ihn berechnen zu lassen, können Sie ein PRINT davorsetzen, dann erhalten Sie gleich den vom Apple errechneten Wert ausgeschrieben.

Sie sollten wirklich diese Beispiele hier mitmachen, es sei denn, Sie haben viel Erfahrung mit der Berechnung von Ausdrücken auf einem Computer. Machen Sie nun nicht gleich alle auf einmal; machen Sie lieber einem nach dem anderen, und prüfen Sie immer zwischendurch nach, wie der Computer den jeweiligen Ausdruck berechnet. Danach ebenso mit dem nächste Beispiel, und so weiter.

```
3 + 2
4 + 6 − 2 + 1
8 * 4
4 ^ 2 + 1
6 / 4 + 1
5 − 4 / 2
4 / 2 − 2
6 * −2 + 6 / 3 + 8
4 + −2
2 ^ 2 ^ 3 + 1
2 * 2 * 3 + 1
2 * 2 + 1 * 3
2 * 2 * 1 + 3
8 / 2 / 2 /1
8 * 2 / 2 / 3 * 2 ^ 2 * 1
20 / 2 * 5
```

Wir geben Ihnen keine der Antworten, denn Ihr Apple wird die richtigen schon für Sie berechnen.

Wie man Vorrangigkeit abändert

Nehmen wir an, Sie wollten 12 durch vier-plus-zwei dividieren.
Wenn Sie dazu schreiben

12 / 4 + 2

erhalten Sie aber zwölf-durch-vier gefolgt von einer Addition von zwei. Aber das wollten Sie nicht. Um also zu erreichen, was Sie sich vorgenommen haben, schreiben Sie

12 / (4 + 2)

Die Klammern haben die Vorrangigkeit im Ausdruck verändert.
Dabei befolgt der Computer eine ganz einfache Regel : ein eingeklammerter Ausdruck wird zuerst berechnet. Und wenn in einem geklammerten Ausdruck ein Unterausdruck geschachtelt eingeklammert ist, wird der innerste dieser geschachtelten Ausdrücke zuerst berechnet. Zum Beispiel :

12 / (3 + (1 + 2) ^ 2)

Hier wird der innerste geklammerte Ausdruck, 1 + 2, zuerst berechnet. Darum erhält man also

12 / (3 + 3 ^ 2)

Nun wissen wir aber, daß 3 ^ 2, also 3 * 3, gleich 9 ist. Somit ist der Wert des geklammerten Ausdrucks 9 + 3, also 12. Der Gesamtausdruck ist mithin einfach 12 / 12, ist also gleich 1.

In einem Fall wie zum Beispiel (9 + 4) * (1 + 2), wo zwar mehr als ein geklammerter Ausdruck vorliegt, aber die Ausdrücke nicht geschachtelt sind, wird wieder von links nach rechts gearbeitet.
Sie erhalten also zunächst 13 * (1 + 2), dann 13 * 3, endlich das Resultat 39.

Wir haben hier noch weitere Ausdrücke mit Klammern für Sie. Es sei denn, Sie sind mit Computern sehr vertraut, lohnt es sich auch hier für Sie sehr, ein paar Minuten zur Errechnung der Ausdrücke aufzuwenden. Der Lohn Ihrer Mühe wird darin bestehen, daß Ihnen später Computerarbeit rasch von der Hand gehen wird.
Übrigens, die meisten vernünftig konstruierten Computersysteme rund um die Welt halten sich an diese Vorrangigkeitsregeln, nicht bloß der Apple.

44 / (2 + 2)
(44 / 2) + 2
3 + (−2 * 2)
(3 + − 2) * 2
100 / 200 / (1 * (9 − 5)))
32 / (1 + (7/3) + (5/4))

KAPITEL 3
Elementare Programmierung

Aufgeschobene Ausführung

Nein, dieses Kapitel ist nicht für Faulenzer gedacht, es handelt sich hier nur um folgendes. Wir schrieben bislang

PRINT 3 + 4

und drückten dann die RETURN -Taste. Daraufhin hat das der Computer sofort getan. Wenn ein Computer einen eingegebenen Befehl sogleich ausführt, dann redet man von **sofortiger Ausführung**. Also haben wir den Apple bisher dazu benutzt, die eingetippten Befehle gleich auszuführen.

Wir wollen jetzt lernen, wie man Befehle für spätere Ausführungen abspeichern kann. Wir nennen das **aufgeschobene Ausführung.** Um sicher zu gehen, daß der Speicher des Computers völlig leer ist, schreiben Sie.

NEW

Wie fast alles, was wir schon kennenlernten, wird NEW von einem Druck auf die RETURN-Taste gefolgt. Um dem Computer nun klar zu machen, daß ein Befehl gespeichert werden soll, setzen wir einfach eine Zahl vor den Befehl. Zum Beispiel, wenn wir

100 PRINT 3 + 4

schreiben, scheint nichts zu passieren, sogar beim Druck der RETURN-Taste. Doch der Apple hat den Befehl gespeichert. Um zu sehen, daß es auch wirklich geschehen ist, schreiben Sie den Befehl

LIST

Versuchen Sie es. Falls Sie sich nicht vertippt haben, (und dann ist wahrscheinlich ein

?SYNTAX ERROR

bei Ihren Bemühungen herausgekommen), so sollte

100 PRINT 3 + 4

Auf dem Bildschirm erscheinen. Schreiben Sie jetzt

RUN

und die Antwort

7

wird auf dem Bildschirm erscheinen.

Der RUN Befehl bewirkt, daß der gespeicherte Befehl nun ausgeführt wird. Dabei wird der Befehl selbst nicht vergessen. Sie können ihn also mit RUN, so oft Sie wollen, erneut ausführen lassen. Überzeugen Sie sich!

Auch wenn Sie nun Bildschirm löschen, wird der gespeicherte Befehl nicht vergessen. Wir lernen an dieser Stelle gleich eine neue Art, den Bildschirm zu löschen :

HOME

der HOME-Befehl hat den gleichen Effekt wie die Tastenkombination,



die wir früher gelernt haben, jedoch kann man HOME auch als aufgeschobenen Befehl benutzen. Um es auszuprobieren, schreiben Sie

```
100 HOME
```

und wenn Sie nun

```
RUN
```

schreiben, so wird es gewissenhaft ausgeführt, und der Schirm wird gelöscht. Schreiben Sie jetzt

```
NEW
```

und dann

```
LIST
```

und beobachten Sie, was geschieht. NEW hat bewirkt, daß der gespeicherte Befehl unwiderruflich gelöscht wurde. Schreiben Sie

```
RUN
```

Nichts passiert. Das liegt daran, daß der alte Befehl mit dem NEW Befehl gelöscht worden ist.

Man kann auch mehrere Befehle speichern, sofern man sie mit verschiedenen Nummern versieht. Etwa

```
1  PRINT "HELLO"
2  PRINT 4 ^ 5
3  PRINT 67/12
```

Zunächst passiert noch nicht viel. Aber schreiben Sie nun

```
RUN
```

und sehen Sie, wie die Antworten erscheinen!



Wir haben also Befehle mit Zahlen versehen, damit sie der Computer abspeichert. Diese Zahlen heißen **Zeilennummern.** Der Computer speichert Befehle in aufsteigender Zeilennummernordnung und führt sie auch in dieser Reihenfolge aus. Um das genau zu verfolgen, löschen wir zunächst die abgespeicherten Befehle mit dem

```
NEW
```

Befehl und schreiben nun :

```
1  PRINT "P"
Ø  PRINT "A"
3  PRINT "E"
2  PRINT "L"
```

Beachten Sie, daß die Null als Zeilennummer erblaubt ist. Die größte verwendbare Zeilennummer ist 63999. Nun führen wir die eben abgespeicherten Befehle mit RUN aus. Dabei sollte das Folgende zu sehen sein :

```
A
P
L
E
```

Um zu prüfen, was im Inneren des Computers gespeichert ist, schreiben Sie

```
LIST
```

Beachten Sie, daß man den LIST Befehl nicht unbedingt vor dem RUN Befehl geben muß. Es ist trotzdem eine gute Idee, mit LIST nachzuschauen, ob man sich verschrieben hat.

Eine Anzahl von Befehlen, die durch RUN ausgeführt werden, heißt **Programm**.

Unser Beispielprogramm hätte

```
A
P
P
L
E
```

drucken sollen. Statt dessen scheint es, als ob dazu ein Befehl fehlt. Wie können wir ihn hinzufügen! Nur dadurch, daß wir Befehle 2 und 3 als Befehle 3 und 4 neu schreiben und einen weiteren mit der Nummer 2 hinzutun. Wir schreiben also :

```
2  PRINT "P"
3  PRINT "L"
4  PRINT "E"
```

Um zu sehen, was passiert ist, tippen Sie LIST.

Beachten Sie, daß der Apple immer die Befehle in der Reihenfolge ihrer Nummern ausführt, egal in welcher Reihenfolge Sie sie geschrieben haben.

Es war lästig, einige Befehle neu zu schreiben, nur damit man einen neuen Befehl einfügen konnte. Es ist daher eine gute Idee, die Zeilennummern beim Programmieren so zu wählen, daß man später Platz hat, falls andere Befehle eingefügt werden sollen. Auch bei der Wahl der ersten Zeilennummer sollte Platz gelassen werden, falls später Befehle ganz zu Anfang eingefügt sollen. Schreiben Sie jetzt

```
NEW
```

um das Programm zu löschen, und speichern Sie das Folgende ab :

```
100  PRINT "C"
110  PRINT "T"
```

Wenn Sie das Programm durch RUN laufen lassen (so nennt man die Ausführung auch), so sehen Sie, daß es nicht ganz das Wort „CAT" vertikal ausdruckt. Aber SIe können jetzt ja

```
105  PRINT "A"
```

später schreiben, um das zu berichtigen. Versuchen Sie es mit LIST und RUN und sehen Sie, was passiert ist. Von nun an werden wir bei allen Programmen mit einer genügend großen Zeilennummer beginnen und aufeinander folgende Zeilennummern in genügend großen Abständen wählen. Dann haben wir ausreichend Platz, falls später weitere Befehle eingefügt werden sollen.

Elementare Aufbereitung

An anderer Stelle besprachen wir schon den Befehl

```
PRINT PDL (0)
```

Damit wurde eine Zahl geschrieben, die uns Aufschluß über die Position einer der Spielkontrollen gab. Man brauchte schon eine ganze Menge von PRINT Befehlen, um vielüber die Kontrollen zu erfahren. Aber jetzt, wo wir Programme schreiben können, wird das Leben viel einfacher. Löschen Sie den Speicher mittels

```
NEW
```

und schreiben Sie

```
100  PRINT PDL (0)
```

Nun wird dieses kurze Programm, immer wenn Sie RUN sagen, ausgeführt. So können Sie jetzt die Stellung der Spielkontrolle leicht sehen.

Befehle abzuspeichern und später mit RUN auszuführen, wird uns Arbeit ersparen, wenn die Befehle mehr als einmal ausgeführt werden sollen. Vorher mußte man den gesamten Befehl oder die Gruppe von Befehlen neu schreiben. Jetzt schreibt man nur

```
RUN
```

Die aufgeschobenen Ausführung bietet aber auch einen anderen Vorteil. Wir können einen Teil des Programms ändern und den Rest beibehalten, ohne daß alles neu geschrieben werden muß. Zum Beispiel :

```
NEW
200  P = PDL (0)
210  PRINT P
220  PRINT " BEWEGEN SIE DIE SPIELKONTROLLE"
230  PRINT "AN EINE NEUE POSITION"
```

Lassen Sie dieses Programm mehrmals laufen, während Sie zwischendurch die Spielkontrollen verändern. Prüfen Sie, ob das Programm auf beide Spielkontrollen reagiert. Es sollte nur auf eine der beiden reagieren. Benutzen Sie diese Gelegenheit, um diejenige Spielkontrolle, deren Stellung das Programm registriert, mit einer Null zu markieren.

Durch eine kleine Veränderung wird das Programm die Stellung der anderen Spielkontrolle schreiben. Listen Sie das Programm so, wie es jetzt ist (LIST), und schreiben Sie

```
200  P = PDL (1)
```

Wenn man einen neuen Befehl eingibt und mit derselben Zeilennummer versieht wie ein schon abgespeicherter Befehl, dann wird der alte Befehl durch den neuen ersetzt. Listen Sie das Programm, um dies zu beobachten. Bewegen Sie nun die andere Spielkontrolle während der Programmabläufe. Reagiert das Programm jetzt auf beide Kontrollen? Markieren Sie die Kontrolle, auf die das Programm reagiert, mit eins.

Diese Art, ein Programm zu ändern, ist ein Beispiel der **Programmaufbereitung.** Was wir gerade lernten, können wir auch dazu verwenden, eine einzelne Zeile zu löschen. Wenn wir Zeile 230 nicht mehr im Programm haben wollen, können wir

```
230
```

schreiben und dann die RETURN-Taste drücken. Dann wird Zeile 230 gelöscht.

Wir hätten auch den DEL-Befehl benutzen können. Um Zeile 230 aus dem Programm zu löschen, würden wir

```
DEL 230, 230
```

schreiben. Die Vorteile des DEL-Befehles werden erst dann sichtbar, wenn wir ganze Abschnitte löschen wollen. Etwa

```
DEL 200, 230
```

löscht jede Zeile mit einer Zeilennummer zwischen 200 und 230 (einschließlich). Probieren Sie das einmal aus, und prüfen Sie mit dem LIST Befehl nach, was genau passiert ist. Dieser Befehl wird uns nützlich werden, wenn wir große Programme schreiben.

Wie wir gesehen haben, gibt es mehrere Befehle, die uns bei der Programmierung helfen. Wir haben

```
NEW
```

zum Löschen des gesamten Programms,

```
LIST
```

zum Listen des Programms, und

```
RUN
```

zum Ausführen des Programms. RUN beginnt mit dem Befehl, der die kleinste Zeilennummer hat. Man kann auch mit der Ausführung woanders anfangen. Ebenso ist es möglich, nur Teile des Programms zu listen. Diese Fähigkeiten werden wir später besprechen.

Elementare Kunstflüge

In diesem Kapitelchen werden wir mit Kunstflügen anfangen und besprechen, wie Schleifen zu machen sind.

Wenn man einen neuen Befehl eingibt und mit derselben Zeilennummer versieht, wie ein schon abgespeicherter Befehl, dann wird der alte Befehl durch den neuen ersetzt. Schreiben wir doch einfach mal das Folgende (natürlich erst nach einem NEW Befehl, denn da könnte ja noch ein altes Programm im Speicher sein):

```
110  PRINT PDL (0)
120  GOTO 110
```

Zeile 110 dieses Programms schreibt die Zahl, die den gegenwärtigen Stand der Spielkontrolle angibt. Zeile 120 bewirkt genau das, was sie zu besagen scheint: die Programmausführung „geht nach" Zeile 110. Und was passiert dann? Das Programm schreibt wieder eine Zahl, die den gegenwärtigen Stand der Spielkontrolle angibt. Danach wird Zeile 120 ausgeführt, die ja besagt, man solle als nächste Zeile die Zeile 110 ausführen. Das wird ohne Ende wiederholt. So etwas nennt man eine **Schleife.** Eine Schleife ist eine Programmstruktur, die einen Befehl enthält, der die erneute Ausführung einer schon vorher ausgeführten Zeile bewirkt. Lassen Sie das Programm laufen, während Sie die Spielkontrolle manipulieren. Im nächsten Abschnitt werden wir Ihnen sagen, wie man dieses Programm anhält. Bewundern wir in der Zwischenzeit, daß der Apple den Befehl PRINT PDL (0) schon einige hundertmal ausgeführt hat - d.h., wenn Sie wirklich vorhin RUN eigegeben haben. Die Fähigkeiten eines abgespeicherten Programms werden nun gewaltig über das hinauswachsen, was man mit unmittelbarer Ausführung schaffen könnte. Und jetzt, wo wir eine gute Grundlage geschaffen haben, wird Ihr Können auf dem Computer in den nächsten Kapiteln dramatisch wachsen!

Weitere Lebenserleichterungen

Zunächst werden Sie sich fragen, wie man wohl unser Programm anhalten kann. Sie haben schon bemerkt, wie die Zahlen auf dem Bildschirm tanzen, wenn Sie die Spielkontrolle bewegen. Das liegt daran, daß die neuen Zahlen unten auf dem Bildschirm geschrieben werden, und während neue geschrieben werden, rücken die alten um eine Zeile nach oben. Das nennt man „aufrollen". Wir haben das schon früher erlebt, natürlich nicht so schnell. Um das Programm zu stoppen, benutzen Sie die folgende Tastenkombination:

CTRL

C

Dieser Befehl gibt Ihnen auch an, an welcher Stelle die Programmausführung gestoppt wurde, indem

```
BREAK IN 110
```

oder eine andere Zeilennummer, bei der das Programm abgebrochen wurde, geschrieben wird. Dieser Befehl ist eine Ausnahme, weil die **RETURN**-Taste nicht abschließend gedrückt wird. Sie dürfen auch die **RESET**-Taste drücken, um das Programm abzubrechen. Dann wird allerdings nicht angegeben, in welcher Zeile das Programm abgebrochen wurde.

Wenn Sie ein Programm auf eine dieser Arten abbrechen, können Sie es mit dem Befehl.

```
CONT
```

fortführen, der „weitermachen" (continue) bedeutet. Versuchen Sie es, und versuchen Sie dann das folgende Programm:

```
NEW
100 X = PDL (0)
110  PRINT "SPIELKONTROLLE NULL IST"
120  PRINT X
130 Y = PDL (1)
140  PRINT "UND SPIELKONTROLLE EINS IST"
150  PRINT Y
```

Wir sagten schon, daß beim Befehl RUN das Programm mit der kleinsten Zeilennummer beginnt. Das stimmt. Wenn Sie aber mit einer anderen Zeile beginnen möchten, etwa Nr. 130 dann schreiben Sie einfach

```
RUN 130
```

Sie können auch für den LIST-Befehl Zeilennummer angeben. Bei

```
LIST 130
```

wird der Apple nur die Zeile 130 listen (natürlich nur, falls diese Zeile existiert), und bei

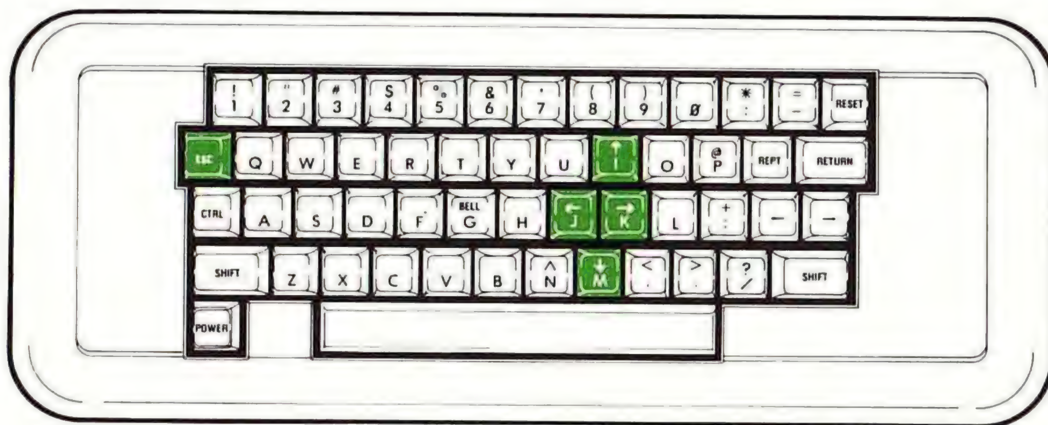
```
LIST 110, 130
```

wird der Apple die Zeilen von Nr. 110 bis Nr. 130 schreiben. Bei dem RUN-Befehl geht eine solche Bereichsangabe aber nicht.

Wie mittels Cursor-Bewegung kopiert oder gelöscht wird

Wenn Sie die Tasten BACKSPACE und RETYPE drücken, dann bewegen Sie den Cursor. Außerdem löschen oder kopieren Sie Buchstaben, wie wir in Kapitel 2 lernten. Es ist aber auch möglich, den Cursor zu bewegen, ohne daß dabei Text verändert wird. Man tut das durch reine Cursorbewegungen.

Fünf Tasten werden dabei benutzt : **ESC**, **I**, **J**, **K** und **M**. Und so wird's gemacht :



Zunächst versetzen wir den Apple in Aufbereitungsmodus, indem wir die Taste **ESC** drücken. Danach benutzen wir **I**, um den Cursor nach oben zu bewegen, **J** für Linksbewegung, **K** für rechts und **M** für nach unten. Um den Cursor eine größere Strecke zu bewegen, drücken Sie gleichzeitig eine der Richtungstasten **I**, **J**, **K**, oder **M** und die **REPT**-Taste. Der Cursor wird dann über den Schirm flitzen solange die Tasten gedrückt bleiben. Wenn der Cursor die obere Bildschirmkante erreicht, bleibt er stehen. Wenn er die untere Kante erreicht, so wird er dort zwar anhalten, aber die Zeilen des Schirms werden nach oben wandern, eine nach der anderen. Wenn der Cursor die rechte Schirmkante erreicht, wird er zunächst verschwinden und dann vorn auf der nächsten Zeile erscheinen (wrap around). Üben Sie diese Cursor-Bewegungen.

Wenn Sie es leid werden, drücken Sie einfach die Leerstaste, dann sind Sie wieder im Normalmodus.

Der Effekt der reinen Cursor Bewegungen mit der **J** und **K**-Taste gleicht scheinbar dem Effekt der BACKSPACE und der RETYPE -Taste, jedenfalls sieht es auf dem Bildschirm so aus. Aber dennoch besteht ein Unterschied, wie Sie mit dem Befehl LIST nachprüfen können. Denn die reinen Cursor-Bewegungen verändern den Text nicht, während die BACKSPACE und RETYPE-Tasten den Text, über den sie sich bewegen, auslöschen oder neu schreiben.

Die reinen Cursorbewegungen haben es aber trotzdem in sich, so daß sie doch nicht so ganz rein und ohne sind. Zum Beispiel schreiben Sie

```
130 PRUNT „DAS WESEN DER GNADE"
140 PRINT „STRENGT SICH NICHT AN"
```

mit einem **RETURN** nach jeder Zeile. Der Apple scheint diese Zeilen zu akzeptieren, aber wenn Sie den RUN Befehl geben, heißt es plötzlich

```
?SYNTAX ERROR IN 130
```

Um die nötigen Korrekturen zu machen, können wir mit folgendem Trick arbeiten: Zuerst listen wir das Programm, dann drücken wir **ESC**. Nun bringen wir mit der Taste **I** den Cursor in die fehlerhafte Zeile und mit **J** zum Anfang der Zeile. Unter Verwendung der RETYPE-Taste schreiben wir alle Buchstaben vor dem U neu und überschreiben das U in PRUNT mit I. Danach gehen wir mit der RETYPE-Taste über die restlichen Buchstaben bis zum Ende der Zeile und drücken dann **RETURN**. Listen Sie nun das Programm, um zu sehen, was passiert ist. Der Computer ist bei Tippfehlern gnädig.

Wenn man einen Teil einer Zeile irgendwo auf dem Bildschirm neu schreiben muß, erleichtern die reinen Cursorbewegung das schon. Einige Minuten Spielerei mit diesen Dingen wird Ihnen später viel Zeit sparen.



Die BACKSPACE-Taste funktioniert nur auf der Zeile, in der Sie gerade schreiben. Wenn sie eine Zeile schreiben und den Cursor bewegen, bevor Sie **RETURN** gedrückt haben, so bewirkt die BACKSPACE-Taste, daß die Programmezeile, die gerade geschrieben wurde, verändert wird, also nicht die Buchstaben, über die sich der Cursor bewegt. Mit der RETYPE-Taste werden allerdings immer diejenigen Buchstaben verändert, über die sich der Cursor bewegt.

Eine Bemerkung zum Lernen von Applesoft BASIC

Oft werden Sie Fragen zu Applesoft BASIC haben, die nicht direkt in diesem Büchlein beantwortet werden. Zum Beispiel in Befehl

PRINT "HELLO"

muß man da einen Zwischenraum zwischen PRINT und „HELLO" lassen? Statt Ihnen diese Frage zu beantworten, empfehlen wir Ihnen, daß Sie Ihren Apple einschalten und es ausprobieren! Oft wird ein solches Experiment Ihre Frage beantworten und Sie werden sich dann später besser daran erinnern können, als wenn Sie die Antwort bloß gelesen hätten.

Gleich passiert ein Unfall

Etwas früher lernten wir, daß man eine Zeile löschen kann, indem man ihre Nummer schreibt und dann **RETURN** drückt. Dies ist eine klassische Methode, Fehler im Programm zu machen. Nehmen wir an, Sie wollten Zeile 1100 vom Programm löschen, aber Sie paßten nicht ganz auf und schrieben

110

gefolgt von **RETURN** Oha! Sie haben gerade Zeile 110 gelöscht. So etwas kann schon mal vorkommen. Oder Sie wollen Zeile 450 verbessern und schreiben in Gedanken

450

Aber dann denken Sie ein bißchen nach und entscheiden sich, die Zeile doch nicht zu ändern. Drücken Sie ja nicht **RETURN**! Gehen Sie entweder mit BACKSPACE zurück über die 450, oder benutzen Sie den besonderen Befehl

CTRL

X

Die Kombination **CTRL-X** setzt ein „/" an das Ende der gerade geschriebenen Zeile und bewirkt, daß die Zeile ignoriert wird. Das heißt, der Apple verhält sich so, als ob Sie die Zeile nicht geschrieben hätten.



Über die Wahrheit...

Apple kann richtig und falsch unterscheiden (true und false). Da das mehr ist, als die meistens von uns können, sollten wir vielleicht etwas darüber sagen. Das Symbol „>" heißt „größer als". Die Behauptung 6 > 2 (also : 6 ist größer als 2) ist sicherlich wahr. Der Apple benutzt die 1 um „wahr" anzuzeigen.

Wenn Sie also schreiben

```
PRINT 6 > 2
```

Dann wird der Computer mit 1 antworten. Die Behauptung 55 > 78 ist natürlich falsch und Apple wird das mit Ø anzeigen. Wenn Sie also schreiben

```
PRINT 55 > 78
```

so erhalten Sie als Antwort Ø.

Das Symbol „>" bedeutet „kleiner als" und man kann auch mit ihm Behauptungen aufstellen. Hier ist eine Liste der Symbole, mit denen Sie Behauptungen aufstellen können.

>	größer als
<	kleiner als
=	gleich
>=	größer oder gleich
<=	kleiner oder gleich
< >	ungleich

Um das Symbol „größer oder gleich" auf dem Apple zu schreiben, drückt man zuerst die > Taste und dann die = Taste. Dasselbe gilt für „kleiner oder gleich" und für „ungleich", was auch durch zweimaliges aufeinander folgendes Tastendrücken geschrieben wird.

Denken Sie nach und prüfen Sie dann auf Ihrem Apple, was bei den folgenden Behauptungen herauskommt.

```
5 < > 5
6 > 2
8 > 8
8 < = 8
9534 = 4359
5 < 8
45 > = -4
-8 < -7
-2 > = -5
9 < > -9
```

In Behauptungen dürfen Sie auch Variablen und Ausdrücke verwenden, nicht nur Zahlen. Etwa

```
PRINT (45 * 6) < > (45 + 6)
```

wird mit 1 beantwortet, da ja 27Ø ungleich 51 ist (wir ernnern uns 1 bedeutet „wahr").

Sie haben schon gesehen, daß der Apple Wahrheit und Lüge trennen kann, jedenfalls wenn es um einfache Zahlen geht. Jedoch eine Behauptung wie ABLE > BAKER kann richtig oder falsch sein, je nachdem welche Werte die Variablen haben. Wenn nun

ABLE = 5
und
BAKER = 9
dann ist die Behauptung
ABLE > BAKER
unrichtig. Aber wenn
ABLE = – 8
und
BAKER = – 15
dann ist die Behauptung
ABLE > BAKER
richtig.

Behauptungen haben also den numerischen Wert von Ø oder 1. Man kann sie in arithmetischen Ausdrücken statt Ø 1 verwenden. Zum Beispiel

```
PRINT 3 + (4 > 2)
```

wird den Wert 4 drucken.

Der Befehl

```
T = 4 < > 3
```

weist T den Wert 1 zu, da 4 ungleich 3 ist und daher 4 < > 3 den Wert 1 hat. Der Befehl

```
HOT = 67 = 19
```

sieht auf den ersten Blick sehr verwirrend aus, aber man kann ihn doch leicht verstehen : Da 67 ungleich 19 ist, ist die Behauptung 67 = 19 falsch, hat also den Wert Ø. Dieser Wert wird dann der Variablen „HOT" zugewiesen.

Wie wir gesehen haben, benutzt der Apple die 1 für „richtig" und die Ø für „falsch". Wenn etwas nicht richtig ist, dann ist es eben falsch. Das mag im Alltag wohl nicht immer so einfach sein, aber der Apple hält sich daran, wie alle Computer. Versuchen Sie

```
PRINT NOT 1
```

und dann

```
PRINT NOT Ø
```

Der Computer stimmt ihnen zu : „nicht (NOT) richtig" ist falsch, „nicht falsch" ist richtig. Natürlich darf man auch Ausdrücke statt Zahlen benutzen. Zum Beispiel

```
PRINT NOT (45 > 3)
```

Der Satz
DREIECKE HABEN DREI SEITEN.
ist richtig. Und der Satz
DIESES BUCH IST AUF DEUTSCH
ist es auch. Überlegen Sie nun
DREIECKE HABEN DREI SEITEN **UND** DIESES BUCH IST AUF DEUTSCH.
Ist das richtig oder falsch? Oh ja, das ist richtig!
Und nun dieser Satz :
DREIECKE HABEN ACHT SEITEN **UND** DIESES BUCH IST AUF DEUTSCH.

Als ganzer ist dieser Satz falsch. Und dieser Satz :
DREIECKE HABEN ACHT SEITEN **UND** DIESES BUCH IST AUF RUSSICH.
Dieser Satz ist auch falsch. Ganz allgemein gilt folgendes : Wenn wir zwei Sätze oder Behauptungen mit UND (auf englisch wäre das AND) verknüpfen, dann gilt :

a) Der neue Satz ist richtig, wenn beide Teilsätze richtig sind.

b) Der neue Satz ist falsch, wenn wenigstens einer der beiden Teilsätze falsch ist.

Der Apple weiß, wie Behauptungen, die mit AND verknüpft wurden, auf ihre Richtigkeit geprüft werden können. Geben Sie Ihrem Computer ein Examen mit dem Folgenden

```
PRINT  1 AND 1
PRINT  1  AND  Ø
PRINT  Ø  AND  1
PRINT  Ø  AND  Ø
PRINT  (3>2)  AND  Ø
PRINT  (NOT Ø)  AND  C4<=5)
```

Ist nun der folgende Satz richtig oder falsch?
EIN DREIECK HAT DREI SEITEN ODER DIESES BUCH IST AUF LATEINISCH!
Er ist richtig. Denn ein Dreieck hat drei Kanten, obwohl dieses Buch nicht auf lateinisch ist. Daher ist der Satz als ganzer richtig. Und davon wollten wir uns ja gerade überzeugen. Wenn zwei Behauptungen mit ODER (auf englisch OR) verknüpft werden, so gilt ganz allgemein folgendes :

a) Der neue Satz ist richtig, falls wenigstens einer der Teilsätze richtig ist.

b) Der neue Satz ist falsch, wenn beide Teilsätze falsch sind.

Der Apple kann auch entscheiden, ob ein Satz richtig ist, der mit „oder" konstruiert wurde. Versuchen Sie die folgenden Befehle — nachdem Sie sich überlegt haben, wie die Antworden lauten müßten.

```
PRINT  1  OR  1
PRINT  1  OR  Ø
PRINT  Ø  OR  1
PRINT  Ø  OR  Ø
PRINT  (4<  >5)  OR  (4=5)
PRINT  1  OR  (Ø  AND  1)
PRINT  ((3>4)  OR  (54<337))  AND  (NOT Ø)
```

AND, OR und NOT werden uns im nächsten Abschnitt viel nützen.

Wir haben schon vorhin gelernt, daß der Computer im Befehl

```
PRINT 1 OR Ø
```

die 1 als richtig und die Ø als falsch auffaßt. Versuchen Sie nun

```
PRINT 23 OR Ø
```

und

```
PRINT – 247 AND 32707.61
```

In solchen Ausdrücken faßt der Computer nicht nur 1, sondern jede von Ø verschiedene Zahl als richtig auf. Wenn er dann den „Wert" der Behauptung errechnet, so ist dieser immer nur Ø oder 1.
In der folgenden Tabelle haben wir Ihnen die Vorrangigkeit der Operationen AND, OR und NOT zusammengefaßt. Allerdings empfehlen wir Ihnen, Klammern zu benutzen, damit Ihre Ausdrücke klarer werden.

Vorrangigkeit der Operationszeichen

1. ( )
2. NOT - (Negationszeichen)
3. ^
4. * /
5. > < = >= <= <>
7. AND
8. OR

Der IF-Befehl

Nehmen wir doch mal an, Sie wollten die Zahlen von 1 bis 10 schreiben, je eine Zahl pro Zeile. Natürlich kann man das so machen :

```
NEW
210  PRINT 1
220  PRINT 2
230  PRINT 3
```

und so weiter. Aber da müßte man ja 10 Befehle schreiben. Wenn Sie nun gar die Zahlen von 1 bis 200 schreiben wollten, müßten Sie 200 Befehle schreiben. Mit unseren bisherigen Kenntnissen können wir die Zahlen von 1 aufwärts mit gerade vier Befehlen schreiben lassen :

```
200  N = 1
210  PRINT N
220  N = N + 1
230  GOTO 210
```

Es gibt nun eine Möglichkeit, festzulegen, wie lange eine solche Schleife wiederholt werden soll. Wir brauchten dazu einen Befehl, wie etwa „GOTO falls N (zum Beispiel) kleiner als 11 ist". Das geht mit dem IF-Befehl. Falls eine Bedingung nicht erfüllt ist, wird der Computer den GOTO-Befehl überschlagen und mit der nächsten Zeile weitermachen. Wenn es keine nächste Zeile gibt, dann hört der Computer einfach auf.

Wir haben hier ein Programm, das von 1 bis 10 zählt, und dann aufhört :

```
200  N = 1
210  PRINT N
220  N = N + 1
230  IF N < 10 THEN GOTO 210
```

Ganz allgemein sieht der IF Befehl so aus :

IF arithmetischer Ausdruck THEN irgendein Befehl

Zuerst wird der arithmetische Ausdruck berechnet. Wenn 0 dabei herauskommt (d.h. „falsch"), wird der Rest des Befehls überschlagen und der Computer macht mit der nächsten Zeile weiter. Wenn der Wert des arithmetischen Ausdrucks nicht 0 ist (d.h. „richtig"), wird derjenige Befehl ausgeführt, der dem THEN folgt.

Der IF-Befehl ist sehr wichtig und wird wohl in fast jedem Programm vorkommen. Nur so zum Spaß versuchen Sie mal

```
NEW
400  GR
410  ROW = 1
420  COLOR = ROW
430  HLIN 0,39 AT ROW
440  ROW = ROW + 1
450  IF ROW < 16 THEN GOTO 420
```

Wie man ein Programm auf der Floppy aufbewahrt

(Überschlagen Sie diesen Abschnitt, wenn Sie keine Floppy haben.)

Jetzt sind Sie vielleicht so weit, daß Sie einige Programme auf der Floppy speichern wollen. Zunächst schreiben Sie, wie bischer, das Programm nach dem NEW-Befehl. Dann schreiben Sie

```
SAVE
```

gefolgt von einem Namen, den Sie benutzen wollen, wenn Sie später das Programm brauchen. Wie immer drücken Sie zum Abschluß die RETURN-Taste. Zum Beispiel, wenn Sie obiges Programm speichern wollen und es STRIPES (Streifen) nennen wollen, würden Sie schreiben

```
SAVE STRIPES
```

gefolgt von RETURN. Daraufhin wird das Programm auf der Diskette unter dem Namen STRIPES gespeichert. Sobald das geschehen ist, können Sie

```
CATALOG
```

schreiben (gefolgt von RETURN) und so eine Liste aller Programme sehen, die auf der Diskette gespeichert sind, darunter auch Ihr STRIPES! Um dieses Programm später von der Diskette zu laden und laufen zu lassen, schreiben Sie :

```
RUN STRIPES
```

Probieren Sie doch mal den SAVE Befehl mit Ihrem Lieblingsprogramm aus. Wenn Sie sich verschrieben haben (auf so etwas folgt beim Apple ?SYNTAX ERROR), schreiben Sie den Befehl erneut ein.

Manchmal ist es wünschenswert, ein Programm in den Speicher des Apple zu laden, ohne es jedoch laufen zu lassen. Das könnte zum Beispiel erwünscht sein, wenn Sie etwas im Programm ändern wollen. In diesem Fall ist der LOAD-Befehl für Sie das Richtige. Man benutzt ihn wie folgt :

```
LOAD
```

gefolgt vom Namen des Programms, das Sie laden möchten. Wenn Sie also STRIPES laden wollen, so schreiben Sie

```
LOAD STRIPES
```

Sie können nun das Programm ändern und dann die neue Version laufen lassen, die ja jetzt im Speicher, aber nicht auf der Diskette ist. Dazu sagen Sie, wie gehabt,

```
RUN
```

Wenn Sie nicht aufpassen und

```
RUN STRIPES
```

sagen, dann wird die alte Version von der Platte geladen und die neue im Speicher überschrieben. Damit ist dann die neue Version verloren.

Sie können die LOAD und SAVE-Befehle dazu benutzen, Programme von einer Diskette auf eine andere zu schreiben. Dazu machen Sie ein LOAD von der einen Diskette und ein SAVE auf die andere Diskette. Üben Sie diese Befehle.

Wie man ein Programm auf dem Band aufbewahrt

(Wenn Sie kein Tonbandgerät benutzen, überschlagen Sie diesen Abschnitt)

Um auf der Kassette ein Programm für späteren Wiedergebrauch zu speichern, müssen Sie zunächst eine leere Tonbandkassette in Ihr Gerät einlegen und es zurückspulen, damit Ihr Programm später leicht zu finden ist. Drücken Sie nun gleichzeitig die Aufnahme und Abspieltaste. Beide sollten heruntergedrückt bleiben. Sie tippen jetzt in Ihren Apple

```
SAVE
```

Wenn Sie abschließend die RETURN-Taste drücken, so wird der blinkende Cursor vom Bildschirm verschwinden. Nach etwa 10 bis 15 Sekunden wird der Computer piepen, um anzuzeigen, daß er jetzt mit der Aufzeichnung beginnt. Ein weiteres Signal zeigt Ihnen später an, daß die Speicherung beendet ist. Der Cursor wird damit wieder auf dem Schirm erscheinen. Drücken Sie nun die Stopptaste Ihres Tonbandgeräts, spulen Sie das Band zurück, und wir sind mit der Operation fertig. Ihr Programm im Speicher des Apple hat sich bei allem nicht verändert.

Weitere Grafikprogramme

Vorhin hatten wir vier Farbpunkte in die Ecken des Bildschirms gezeichnet. Schreiben Sie nun

```
NEW
190 GR
200 COLOR = 9
210 PLOT 0.0
220 PLOT 0,39
230 PLOT39,39
240 PLOT 39.0
```

Listen Sie das Programm und prüfen Sie, ob Sie es richtig eingegeben haben. Befehlen Sie dann RUN. Es geht schnell, nicht wahr? Um die Farben zu ändern, verändern Sie bloß Zeile 200 und sagen erneut RUN. Versuchen Sie es nun mit LIST.
Beachten Sie, daß die Programmliste hinter einem schmalen Fenster unten auf dem Bildschirm vorbeiläuft. Das wird so bleiben, bis Sie

```
TEXT
```

schreiben, damit kommen Sie aus dem GRafikmodus heraus.

Das folgende Programm wird nun den ganzen Bildschirm mit einer Farbe bemalen.

```
NEW
200  GR
210  COLOR = 9
220  COLUMN = 0
230  VLIN 0,39 AT COLUMN
240  COLUMN = COLUMN + 1
250  IF COLUMN < 40 THEN GOTO 230
```

Wir werden jetzt Schritt für Schritt erklären, was im Einzelnen passiert, wenn dieses Programm läuft. Zeile 200 bringt uns in GRafikmodus. Die Farbe wird in Zeile 210 gewählt. Das Programm fängt nun mit der Spalte 0 des Bildschirm an und arbeitet sich bis Spalte 39 durch.

Zeile 220 sorgt dafür, daß wir in Spalte 0 anfangen. Mit Zeile 230 wird nun eine senkrechte Linie gezeichnet, und zwar in der Spalte 0. Wir sind nun mit Spalte 0 fertig und werden in Zeile 240 die Spaltennummer um eins erhöhnen. Zeile 250 prüft dann, ob der Wert von COLUMN kleiner als 40 ist. Wenn ja, dann „geht es zurück" zur Zeile 230, und eine neue senkrechte Linie wird gezeichnet. Wenn COLUMN aber den Wert 40 erreicht (auf dem Bildschirm des Apple ist die rechts äußere Spalte ja die Spalte 39), geht es nicht wieder zurück. Man sagt dazu auch, daß das Programm „durch das IF durchfällt". Damit hört das Programm hier auf, denn es folgen keine weiteren Programmzeilen.

Um nicht immer RUN sagen zu müssen, wenn wir den Bildschirm einfärben wollen, können wir.

```
260 GOTO 210
```

schreiben. Schauen Sie genau hin, und sehen Sie, was passiert. Wann wird das Programm stoppen? Listen Sie das Programm auf und seien Sie sicher, daß Sie genau verstehen, was da passiert, bevor Sie in diesem Buch weitermachen.

Wenn Sie nun nicht mehr mit diesem Programm weiter herumspielen wollen, löschen Sie den Speicher, und versuchen Sie es mit dem folgenden Programm. Darin wird ein neuer Befehl verwandt : REM. Das ist die Abkürzung von „Remark", was etwa „Anmerkung" bedeutet. Dieser Befehl erlaubt Ihnen, Kommentare in Ihr Programm einzufügen. Der Computer ignoriert diese Kommentare, denn sie sind eben nur für Menschen gedacht. Beachten Sie, um wieviel leichter das folgende Programm zu verstehen ist, nur weil viele REM-Befehle eingefügt wurden.

```
200  REM GRAFIKMODUS SETZEN
210  GR
220  REM FARBE AUSWÄHLEN
230  COLOR = 1
240  REM SPIELKONTROLLE 0 LESEN
250  X = PDL (0)
260  REM DURCH 7 TEILEN, SODASS HÖCHSTER WERT VON X 36 IST
270  X = X/7
```

```
280  REM SPIELKONTROLLE 1 LESEN
290  Y = PDL (1)
300  REM Y TEILEN, SODASS DER WERT AUCH AUF DEM BILDSCHIRM LIEGT
310  Y = Y/7
320  REM DEN PUNKT ZEICHNEN
330  PLOT X, Y
340  GOTO 250
```

Nachdem Sie dann RUN geschrieben haben, drehen Sie doch mal an den Spielkontrollen. Dieses Programm nennt sich „Etch-a-sketch", benannt nach einem Spielzeug in den USA, das ähnlich funktioniert.

Man muß durch 7 teilen, da die PDL-Funktion einen Wert zwischen 0 und 255 liefert, während der Schirm nur mit Koordinaten bis zu 39 arbeiten kann. Division durch 7 liefert also als höchste Werte für X und Y 255/7 = 36.4285715. Im GR-Modus wird das automatisch zu einem Koordinatenwert, und zwar durch **abrunden.** Mit anderen Worten, X und Y werden Werte nur zwischen 0 und 36 haben. Daher erreichen wir nicht den gesamten Bildschirm, weder in Höhe noch in Breite. Um das zu erreichen, müssen wir statt

```
270 X = X/7
```

zwei Zeilen hinschreiben, und zwar

```
270  IF X > 239 THEN X = 239
275  X = X/6
```

Nun erreichen wir die gesamte Breite. Um dann auch die volle Höhe zu schaffen, müssen wir die Berechnung von Y in Zeile 310 auch entsprechend abändern.

Der IF-Befehl beschränkt den X Wert auf 239. In Apples Grafikmodus mit niedriger Auflösung wird 239/6 von 39.833333 auf 39 abgerundet. Diese Verwendung des IF-Befehls kommt oft vor.

FOR / NEXT Schleifen

Wenn man Schleifen macht, so gibt es oben und unten, sowohl beim Fliegen, als auch beim Programmieren. Im Programm

```
NEW
100  NUMBER = 0
110  PRINT NUMBER
120  NUMBER = NUMBER + 1
130  IF NUMBER < = 12 THEN GOTO 110
```

ist Zeile 110 das **obere** Ende und Zeile 130 das **untere** Ende der Schleife. Das Programm schreibt die Zahlen von 0 bis 12 einschließlich. Die Zahl 12 ist der **Grenzwert** der Schleife. Man könnte diese Schleife auch anders schreiben unter Benutzung eines Befehls, mit dem wir uns noch nicht befaßt haben : dem FOR Befehl. Wir schreiben damit das obige Programm folgendermaßen neu :

```
200  FOR NUMBER = 0 TO 12
210  PRINT NUMBER
220  NEXT NUMBER
```

Schreiben Sie

```
RUN 200
```

um dieses Programm zu benutzen. Wenn Sie bloß RUN schreiben, so wird das Programm mit der Zeile 100 ausgeführt (100 ist ja momentan die niedrigste Zeilennummer).

Zeile 200 enthält also den neuen FOR-Befehl. Der Beginnt damit, daß der Variablen NUMBER der Wert 0 zugewiesen wird. Das war genau das, was Zeile 100 gemacht hatte. Dann wird Zeile 210 ausgeführt. Das untere Ende der Schleife ist Zeile 220. Dort wird die Variable NUMBER im Wert um 1 erhöht und dann mit dem Grenzwert, der im FOR gegeben wurde, verglichen, also mit der 12. Wenn der Wert von NUMBER diese Grenze nicht überschritten hat, dann geht es mit der Ausführung weiter, und zwar mit der Zeile, die unmittelbar auf das FOR folgt. Wenn der Variablenwert den Grenzwert überschreitet, dann „fällt das Programm durch" zu der Zeile, die dem NEXT folgt. Da in unserem Programm nichts mehr nach dem NEXT steht, ist das Programm also zu Ende.

Ein offensichtlicher Vorteil der FOR/NEXT Methode ist natürlich, daß man sich einen Befehl spart. Aber noch wichtiger ist, daß man sich mit FOR/NEXT nicht ganz so beim Schleifenschreiben anstrengen muß. Wenn wir waagerechte Linien auf dem Schirm in jeder der 15 Farben zeichnen wollen, könnten wir folgendes schreiben

```
3000  GR
3010  FOR N = 0 TO 15
3020  COLOR = N
3030  HLIN 0,39 AT N
3040  NEXT N
```

Ein weiterer Vorteil muß genannt werden: Es ist viel leichter, ein einzelnes FOR zu lesen, als drei Befehle, von denen man noch herauskriegen muß, was sie nun wirklich tun. Das untere Ende der FOR Schleife findet man, indem man sich bloß das nächste NEXT sucht, das den gleichen Variablenamen trägt wie das FOR.

Wir möchten hinzufügen, daß Sie das FOR nicht unbedingt benutzen müssen. Sie sollten aber verstehen, wie es funktioniert, denn es verleiht uns keine neuen Programmierfähigkeiten, es macht es nur für manche Leute einfacher, ein Programm zu schreiben.

An dieser Stelle sollten Sie die Zeilen 3000 bis 3040 einschließlich löschen, falls Sie bis hierher mitgemacht haben. Schreiben Sie also

```
DEL 3000, 3040
```

Listen Sie gleich das Programm, damit Sie sicher sind, daß alles ist, wie es sein soll.

Um nur die geraden Zahlen von 0 bis 12 zu schreiben, könnten Sie folgendes Programm benutzen:

```
100  THING = 0
110  PRINT THING
120  THING = THING + 2
130  IF THING < = 12 THEN GOTO 110
```

Der Trick passiert in der Zeile 120, wo der Wert von THING um 2 erhöht wird. Die Schleife hat einen **Schritt** von 2 (auf englisch STEP), wie man sagt. Um das im FOR zu erreichen, sagen wir

```
200  FOR THING = 0 TO 12 STEP 2
```

Der Rest des Programms sähe wie die Zeilen 210 und 220 auf der vorherigen Seite aus, außer daß statt NUMBER eben überall THING geschrieben werden muß. Vesuchen Sie das mal. Der Wert, der dem STEP folgt, kann eine beliebige Zahl Zahlenbereich des Apples sein. Man kann sogar rückwärts schreiben, etwa

```
200 FOR THING = 39 TO 15 STEP - 3
```

Schreiben Sie es einmal, und probieren Sie es mit

```
RUN 200
```

aus. Sie sollten überhaupt mit dem FOR Befehl ein bißchen herumspielen, damit Sie dafür ein Gefühl bekommen. In mehreren folgenden Beispielprogrammen werden wir von jetzt ab das FOR verwenden.

Neben den Bequemlichkeiten des FORs gibt es auch einige Einschränkungen, die man beachten muß. Zum Beispiel darf man FOR/NEXT Schleifen schachteln, aber ihre Bereiche dürfen sich nie teilweise überschneiden. Wir werden das mit einigen Beispielen veranschaulichen

```
NEW
300  GR
310  FOR HUE = 1 TO 15
320  COLOR = HUE
330  FOR ROW = 0 TO 39
340  HLIN 0, 39 AT ROW
350  NEXT ROW
360  COLOR = HUE - 1
370  FOR COLUMN = 0 TO 39
380  VLIN 0, AT COLUMN
390  NEXT COLUMN
400  NEXT HUE
```

Dieses Programm verdeutlicht Schleifenschachtelung der Tiefe 2. Meditieren Sie zuerst über das Programm und lassen Sie es dann laufen, bevor Sie weitermachen. Erinnern Sie sich daran, wenn Sie in einem Programm ein FOR schreiben, so müssen Sie auch ein entsprechendes NEXT haben.

Ein falsches Programm

```
NEW
500  FOR N = 10 TO 20
510  PRINT N
520  FOR J = 30 TO 40
530  PRINT J
540  NEXT N
550  NEXT J
```

Diese Programm ist unkorrekt. Die beiden Schleifen überschneiden sich teilweise. Dabei kommt nicht nur eine Fehlermeldung heraus, es ergibt auch ohnehin keinen Sinn. Wenn sich irgenwann Ihre Schleifen überschneiden, hat sich Ihr Denken verheddert. Wenn Sie genau wissen, was Sie machen wollen, und immernoch Ihre Schleifen sich überschneiden lassen wollen, dann sollten Sie lieber die Schleifen mit IF konstruieren. Da können sie sich beliebig überschneiden, was auch immer dabei herauskommen mag.

Ein letztes Beispiel für Schleifenschachtelung

```
NEW
3ØØ  GR
31Ø  HUE = Ø
32Ø  FOR COLUMN = Ø TO 35 STEP 5
33Ø  FOR LINE = Ø TO 3Ø STEP 1Ø
34Ø  HUE = HUE + 1
35Ø  IF HUE > 15 THEN HUE = Ø
36Ø  COLOR = HUE
37Ø  FOR ROW = LINE TO LINE + 9
38Ø  HLIN COLUMN, COLUMN + 4 AT ROW
39Ø  NEXT ROW
4ØØ  NEXT LINE
41Ø  NEXT COLUMN
```

Dieses Programm schachtelt die Schleifen zur Tiefe 3, und es zeichnet Ihnen ein buntes Muster. Beachten Sie, daß COLOR nicht als Variable in einem FOR/NEXT benutzt werden darf. COLOR ist ein reserviertes Wort im Applesoft. Versuchen Sie, das Programm im Textmodus laufen zu lassen, nachdem Sie Zeile 3ØØ entfernen. Befehlen sie RUN. Was passiert nun?

Jetzt wird geblinkt

Wenn es Sie ein bißchen langweilt, immer nur die Buchstaben weiß auf dunklem Hintergrund auf dem Bildschirm anzustarren, werden Sie an diesem Abschnitt Spaß haben. Schreiben Sie mal

```
INVERSE
```

und sehen Sie sich nun Applesoft Bereitschaftszeichen und den Cursor an. Sie sollten schwarz auf weißem Hintergrund erscheinen. Schreiben Sie nun ein einfaches Programm

```
NEW
1ØØ  PRINT "SCHWARZ AUF WEISS"
```

Lassen Sie es mal laufen. Ja, das sieht schon besser aus! Schreiben Sie nun

```
FLASH
```

und lassen Sie das Programm nochmals laufen. Das blinkt aber schön!

Beachten Sie, INVERSE und FLASH beziehen sich nur auf Zeichen, die der Computer schreibt. Was Sie selber eintippen, verändert sich nicht. Man kan diese Befehle sowohl in unmittelbarer Ausführung, als auch bei aufgeschobener Ausführung benutzen. Experimentieren Sie damit. Nachdem Sie ordentlich viel geblinkt und viel schwarz auf weiß geschrieben haben, denken Sie sich vielleicht, daß weiß auf schwarz vielleicht doch nicht so langweilig ist. Wenn Sie also wieder weiß auf schwarz möchten, dann können Sie

```
NORMAL
```

schreiben, dann geht es wieder wie vorher zu.

Charmanter PRINT-Befehl

Als Experiment wollen wir das folgende Programm eingeben und mit RUN herausfinden, was geschieht :

```
NEW
100  PRINT "HELLO"
110  GOTO 100
```

Halten Sie nun das Programm mit CTRL-C. Nun ändern Sie Zeile 100, indem Sie nur ein Komma anfügen

```
100  PRINT "HELLO",
```

Lassen Sie es erneut laufen. Wie Sie sehen, werden die Worte in Spalten geschrieben. Setzen Sie nun ein Semikolon anstatt des Kommas ein

```
100  PRINT "HELLO";
```

Jetzt sehen Sie beim Laufenlassen des Programms, daß die Ausgabe „gepackt wird". Das heißt, zwischen den vom Computer geschriebenen Feldern wird kein Zwischenraum gelassen. Ein HELLO nach dem anderen wird geschrieben, bis bald der ganze Bildschirm voll ist.

Ändern Sie Ihr Programm. Fügen Sie hinzu

```
90 V = 99
```

und ändern Sie Zeile 100 auf

```
100 PRINT V
```

Lassen Sie es laufen. Dann ändern Sie es um auf

```
100 PRINT V,
```

Lassen Sie es erneut laufen, und probieren Sie es dann mit

```
100 PRINT V;
```

Beachten Sie, daß Sie Komma und Semikolon auch bei numerischen Werten verwenden dürfen. Manchmal erweist es sich als sehr nützlich, Zahlen ohne Zwischenraum hintereinander schreiben zu dürfen. Sie können auch Komma und Semikolon innerhalb eines PRINT Befehls verwenden. Löschen Sie das alte Programm mit NEW und schreiben Sie

```
100  STRIKES = 2
110  BALLS = 3
120  PRINT STRIKES, BALLS
```

Man kann die Ausgabe übersichtlicher machen, indem man ihr Kommentare einfügt. Ändern Sie z.B. Zeile 120 um auf

```
120  PRINT " STRIKES UND BALLS SIND", STRIKES, BALLS
```

Beachten Sie, daß Sie wahrscheinlich ein Komma zwischen dem Text und „STRIKES" haben wollen. Oder Sie wollen ein Semikolon. Aber dann brauchen Sie doch wohl ein Leerzeichen nach dem „SIND", sonst wird die folgende Zahl zu nahe an den Text gerückt. Wenn Sie einen großen Zwischenraum zwischen den beiden Zahlen hübsch finden, dann können Sie schreiben :

```
120  PRINT "STRIKES UND BALLS SIND"; STRIKES;" "BALLS
```

In dieser Version steht nur ein Leerzeichen zwischen den beiden Zahlen. Vielleicht die hübscheste Form wird mit dem folgenden erreicht (machen Sie überhaupt auf dem Apple mit?)

```
120  PRINT "STRIKES" ";STRIKES;" BALLS ";BALLS
```

Nun sieht es ganz professionell aus.

Nehmen wir doch einmal an, Sie wollten das Wort „HIER" schreiben, und zwar in der Spalte 10 (Sie wissen ja, der Bildschirm hat 40 Spalten). Sie könnten es so machen

```
120  PRINT "HIER"
```

Glauben Sie dem Experten, es sind schon 9 Leerzeichen vorne weg! Sie können aber auch die TAB Funktion verwenden. Die Funktion ist geradeso wie bei Schreibmaschinen. Der Befehl

```
120  PRINT TAB (10) "HIER"
```

Hat genau den gleichen Effekt, und Sie brauchen dann nicht wie oben die neun Leerzeichen davorsetzen. Versuchen Sie es, es wird Ihnen Freude machen!

Wenn Sie TAB mit dem FOR kombinieren, dann können Sie raffinierte Effekte in der Ausgabe erzielen. Zum Beispiel

```
NEW
200  FOR N = 1 TO 24
210  PRINT TAB (N) "X"
220  NEXT N
```

Es gibt übrigens nur 24 (und nicht 40) Zeilen auf dem Bildschirm. Daher haben wir 24 als Grenzwert für die obige Schleife gewählt. Man kann TAB nicht dazu verwenden, nacht links innerhalb der Zeile einzurücken. Also immer nur vorwärts nach rechts. Um eine bestimmte Zeile zu beschreiben, können Sie die VTAB-Funktion benutzen (vertikaler TAB). Die oberste Zeile des Bildschirm ist die Zeile 1, die unterste ist die Zeile 24. VTAB, anders als TAB, darf nicht im PRINT-Befehl verwandt werden.

Sie können die HTAB Funktion verwenden, wenn Sie nicht den PRINT-Befehl in Verbindung mit TAB verwenden möchten. Mit HTAB darf man nun sowohl nach rechts als auch nach links gehen, wobei die Spalte links außen 1 ist, und die Spalte rechts außen 40. Das Programm auf der nächsten Seite verdeutlicht den Gebrauch von HTAB und VTAB.

```
NEW
590  HOME
600  FOR X = 1 TO 24
610  FOR Y = 1 TO X
620  HTAB X
630  VTAB Y
640  PRINT "APPLE"
650  NEXT Y
660  NEXT X
670  GOTO 600
```

Bevor Sie das Programm laufen lassen, versuchen Sie doch zuerst mal herauszuknobeln, was es eigentlich leistet (es ist nicht ganz so einfach). Sie werden gewiß angenehm überrascht sein, denn die Ausgabe wird schön aussehen.

TAB kann man in sofortiger Ausführung benutzen, aber HTAB und VTAB werden nur in Programmen verwandt. Während es so scheint, als würden TAB, HTAB und VTAB so wie das Koordinatensystem funktionieren, gibt es doch einige Abweichungen. TAB numeriert die Spalten von 1 bis 40, geradeso wie es bei Schreibmaschinen ist. Aber die Koordinaten von PLOT gehen von 0 bis 39, denn es ist etwas bequemer, wenn Sie Grafikprogramme schreiben. Buchstaben sind höher als die ,,Blöcke'', mit denen wir Grafiken zeichnen. Daher haben wir nur für 24 Schreibzeilen auf dem Bildschirm Platz. Also muß man dem VTAB Zahlen zwischen 1 und 24 geben. Wenn Sie sich nicht daran halten und 0 oder eine zu große Zahl benutzen wollen, wird man Ihnen

```
?ILLEGAL QUANTITY ERROR
```

mitteilen. Bei VTAB kann man, wie gesagt, nur bis 24 gehen. Aber bei HTAB und TAB sind Werte bis zu 255 zulässig. Sowohl TAB als auch HTAB können nämlich auf der nächsten Zeile weitermachen. Um das mitzuerleben, schreiben Sie

```
NEW
300  FOR K = 1 TO 255
310  PRINT TAB (K) K
320  NEXT K
```

Ersetzen Sie nun Zeilen 310 und 320 durch

```
310  HTAB K
320  PRINT K
```

undd fügen Sie hinzu

```
330  NEXT K
```

Und was passiert, wenn Sie HTAB durch VTAB im Programm ersetzen?

KAPITEL 4
Alles über Grafiken

Wenn Sie dieses Programm laufen lassen, wird Ihnen das Resultat wahrscheinlich langweilig vorkommen. Sie können aber das Bewegungsmuster verändern, indem sie XOLD und YOLD verändern (Zeile 500 und 510). Wir haben noch einen anderen Vorschlag, der Ihnen besser gefallen wird :

```
580  XNOW = XOLD + XMOVE * PDL(0) / 70
686  YNOW = YOLD + YMOVE * PDL(1) / 70
```

Wenn Sie jetzt an den Spielkontrollen drehen, werden Sie sofort den erzielten Effekt feststellen können.

Und noch einen Vorschlag. Warum lassen wir uns nicht (mittels eines zweiten INPUT-Befehls) eine Hintergrundsfarbe BACKGROUND geben? Wir könnten dann zunächst den Bildschirm mit dieser Hintergrundsfarbe bemalen, bevor wir mit dem Programm anfangen, also gleich nach dem GR Befehl. Um die alte Ballposition auszuwischen, müßten wir einfach Zeile 780 abändern :

```
780  COLOR = BACKGROUND
```

Versuchen Sie es auch mit

```
780  COLOR = BACKGROUND + 3
```

Speichern Sie sich Ihre Lieblingsversion des Programms auf Diskette.

Wie man Töne macht

Sie können ganz einfach Geräusche wie Klicken, Ticken, Rasseln usw. auf Ihrem Apple machen, indem Sie auf das Gehäuse klopfen oder ihn gar vom Tisch werfen. Das ist Ihre Sache. Wir werden uns hier mit Tönen befassen, die man programmieren kann. Gehen Sie also beim Durcharbeiten dieses Abschnitts in eine stille Ecke.

Um mit einem Programm auf dem Apple einen Ton zu erzeugen, brauchen wir die folgende magische Formel :

```
150  SOUND = PEEK (-16336)
```

Für diese Formel gibt es keine einfache Erklärung. Die Zahl –16336 hat damit zu tun, an welcher Adresse im internen Speicher des Apple sich der Lautsprecher „befindet", der zu den elektronischen Komponenten des Computers gehört. Sie müssen die Zahl bei Bedarf schon nachschlagen.

Die Funktion PEEK liefert einen numerischen Wert, der an einer besonderen Stelle im internen Speicher steht. Für die meisten Argumentwerte von PEEK ist das also eine Zahl. Für gewisse andere, etwa –16336, passiert statt dessen etwas. In diesem Fall macht der Lautsprecher einen kleinen Klick. Jedesmal, wenn der Befehl ausgeführt wird, kommt dieser minuziöse Klick zustande. Lassen Sie das obige Miniprogramm laufen und hören Sie genau hin.

Fügen Sie jetzt hinzu

```
160  GOTO 150
```

und lassen Sie das Programm nochmals laufen. Na, das kann man ja nicht überhören!

Damit das Programm nur eine Zeitlang ertönt, könnte man noch schreiben

```
140  FOR BEEP = 1 TO 100
160  NEXT BEEP
```

Versuchen Sie's.

Wir haben einen Ton durch eine schnelle Folge von Klicks erzeugt. Jedes Programm, das PEEK(–16336) wiederholt benutzt, wird irgendeinen Laut produzieren. Da es lästig fällt, ständig –16336 zu schreiben, werden wir noch eine befreiende Kleinigkeit hinzufügen. Schreiben Sie

```
100  S = –16336
```

Um mit Zeile 150 einen resonanten Klick zu machen, verändern Sie sie so

```
150  SOUND = PEEK(S) – PEEK(S) + PEEK(S) – PEEK(S) + PEEK(S) – PEEK(S)
```

Eine unterschiedliche Anzahl von PEEK(S) in der Zeile wird einen Klick unterschiedlicher Qualität erzeugen. Probieren Sie verschiedene Möglichkeiten aus.
Machen Sie Ihre eigenen Versionen. Je schneller die Schleife ist, desto höher fällt der Ton im allgemeinen aus.

Um diese Töne zu verwenden, laden wir das Programm des springenden Balls. Jedesmal, wenn der Ball von der Wand abprallt und seine Richtung ändert, werden wir einen Ton erzeugen.

Eine Lösung ist weiter unten gegeben. Versuchen sie aber zuerst, selbst eine Lösung zu finden. Sie müssen dabei den Ton erzeugen, wenn XMOVE oder YMOVE ihr Vorzeichen verändern.

Der springende Ball macht Geräusche

Hier haben wir eine Version, bei der das Abprallen des springenden Balls hörbar ist.

```
240  REM  S  DEN WERT DES LAUTSPRECHERS ZUWEISEN
260  S = –16336
663  REM ABPRALL GERÄUSCH
665  FOR B = 1 TO 5
670  BOUNCE = PEEK(S) – PEEK(S) + PEEK(S) – PEEK(S)
675  NEXT B
695  REM ABPRALL GERÄUSCH
696  FOR B = 1 TO 5
697  BOUNCE = PEEK(S) – PEEK(S) + PEEK(S) – PEEK(S)
698  NEXT B
```

Versuchen Sie es nun mal mit Ihrer Version. Sie könnten auch für jede der vier Wände einen anderen Ton erzeugen.

Höhere Töne

Um noch höhere Töne zu erhalten, können wir mit einem neuen Aspekt des Applesoft BASIC arbeiten. Man kann nämlich auch mehr als einen Befehl auf dieselbe Zeile schreiben. Versuchen Sie folgendes :

```
NEW
50  S = PEEK(-16336) : GOTO 50
```

Der Doppelpunkt (:) kann benutzt werden, um verschiedenen Befehle auf derselben Zeile voneinander zu trennen. Allerdings trägt dabei nur der erste Befehl eine Zeilennummer, so daß man nur zum ersten Befehl einer jeglichen Zeile mittels GOTO gelangen kann.

Fügen Sie nun hinzu

```
40  FOR PAUSE = 1 TO 2500 : NEXT PAUSE
```

Mehrere Befehle auf eine Zeile zu schreiben hat folgende Vorteile :

1. Die Befehle werden schneller ausgeführt. (Das wird nur dann für Sie wesentlich sein, wenn Sie schnellere Ausführungsgeschwindigkeit benötigen.)
2. Mehr von Ihrem Programm kann gleichzeitig auf dem Bildschirm erscheinen.
3. Sie brauchen weniger zu schreiben.
4. Sie können mehrere Befehle gruppieren, die logisch zusammengehören und gemeinsam eine Funktion ausführen. Das ist der Fall mit der Zeile 40 oben.
5. Man braucht weniger internen Speicher. (Das wird nur dann für Sie wichtig, falls Ihnen der Speicherplatz zu knapp wird, und Sie die Nachricht ?OUT OF MEMORY oder PROGRAM TOO LARGE beim Eintippen Ihres Programms zu Gesicht bekommen.)

Es gibt auch Nachteile :

1. Das Programm ist schwerer zu verstehen.
2. Es wird schwieriger, das Programm zu korrigieren und Fehler zu berichtigen.
3. Man kann mit GOTO nur den ersten Befehl einer Zeile erreichen.
4. Es ist etwas entmutigend, gerade mit einer langen Zeile fertig geworden zu sein, und dann beim RUN ein ?SYNTAX ERROR zu bekommen, so daß die ganze Zeile ganz von vorne wieder getippt werden muß.

Zufallszahlen

Versuchen Sie es mit diesem kurzen Programm auf Ihrem Apple :

```
NEW
100  PRINT RND(1)
110  GOTO 100
RUN
```

RND in Zeile 100 ist die Abkürzung von RaNDom (zufällig). Diese Funktion liefert Zufallszahlen, das heißt, man kann nicht vorhersagen, was als nächste Zahl kommen wird. Halten Sie das Programm mit





an. Die von diesem Programm erzeugten Zahlen waren Dezimalbrüche zwischen null und eins mit einer Zufallsverteilung.

Ändern Sie Zeile 100 um in

```
100  PRINT RND(6)
```

Lassen Sie das Programm erneut laufen.

Halten Sie nun das Programm aufs neue mit CTRL-C an. Ja, interessant. Die Zufallszahlen liegen immer noch zwischen 0 und 1. Machen Sie folgendes Experiment : Schreiben Sie sich die letzte generierte Zahl auf, damit Sie sich an sie erinnern können.

Ändern Sie Zeile 100 erneut :

```
100  PRINT RND(0)
```

Lassen Sie das Programm laufen und vergleichen Sie die Zahl auf dem Bildschirm mit der niedergeschriebenen Zahl. RND(0) liefert die zuletzt erzeugte Zufallszahl.

Dezimale Brüche zwischen eins und null können etwas umständlich sein. Oft kommt man mit ganzen Zahlen, wie 3, 6 und 10, besser zurecht. Um Zufallszahlen zwischen 0 und 9 zu erzeugen, müssen wir dem Programm ein paar Zeilen hinzufügen.

```
NEW
 90  REM  X  ZUFALLSZAHLEN ALS WERT ZUWEISEN
100  X = RND(1)
110  REM X MIT 10 MULTIPLIZIEREN
120  X = X * 10
130  REM NUN DIE STELLEN HINTER DEM KOMMA WEGSCHNEIDEN
140  X = INT(X)
150  PRINT X
160  GOTO 100
```

Zeile 140 enthält die uns neue INT Funktion. INT(X) liefert die größte ganze Zahl, die kleiner oder gleich X ist. Wenn X zum Beispiel den Wert 3.6754 hat, ist INT(X) gleich 3. Die Klammern, die hier das X umschließen, dürfen einen beliebigen arithmetischen Ausdruck oder numerische Variable umschließen.

Lassen Sie das Programm nun laufen. Funktioniert es so, wie Sie es erwartet haben? Um Zahlen zwischen 1 und 10 zu erzeugen, fügen wir

```
145  X = X + 1
```

hinzu. Prüfen Sie es nach!

Auf den ersten Blick mag das Programm etwas kompliziert aussehen. Um Schritt für Schritt mitzuerleben, was genau passiert, könnte man eigens Haufen von PRINT Befehlen einfügen. Ändern Sie also Ihr Programm folgendermaßen. :

```
 90  REM  X  ZUFALLSZAHL ZUWEISEN
100  X = RND(1) : PRINT "X = RND(1)",X
105  PRINT
110  REM X MIT 10 MULTIPLIZIEREN
120  X = X * 10 : PRINT "X = X*10",X
125  PRINT
130  REM ABRUNDEN
140  X = INT(X) : PRINT "X = INT(X)",X
145  PRINT
150  REM X UM 1 ERHÖHEN
160  X = X + 1 : PRINT "X = X+1"
170  PRINT X
175  PRINT
180  FOR PAUSE = 1 TO 2000 : NEXT PAUSE
190  GOTO 100
```

Lassen Sie dieses Programm laufen, um den Effekt zu verfolgen.

Wenn Sie raffiniert sein wollen, können Sie auch dieses Programm auf nur eine Zeile bringen :

```
100  PRINT INT (10 * RND(1)) + 1 : GOTO 100
```

Verstehen Sie, wie Zeile 100 funktioniert?

Wie man zwei Würfel nachmachen kann

Was wir bis hierher über Zufallszahlen gelernt haben, können wir dazu verwenden, um ein Programm würfeln zu lassen. Und zwar

```
NEW
100  PRINT "WEISSER WÜRFEL",
110  PRINT INT (6 * RND(1)) + 1
120  PRINT "ROTER WÜRFEL",
130  PRINT INT (6 * RND(1)) + 1
```

Dieses Programm erzeugt Zufallszahlen zwischen 1 und 6, gerade wie es ein Würfel tut. Um den weißen und den roten Würfel zu werfen, sagen Sie RUN. Versuchen Sie doch mal, diese Würfel in ein Spiel mit einzubeziehen und ein Programm dafür zu schreiben!

Versuchen Sie auch, mit einer einzigen Programmzeile Zufallszahlen zwischen 1 und 50 zu erzeugen, desgleichen für einen Bereich zwischen 0 und 25. Danach versuchen Sie's mit selbstgewählten Zahlen. Vergessen Sie nicht, eine 1 zu addieren, falls die 0 unerwünscht ist.

Hier ist ein Programm, das Zufallszahlen in Farbe überträgt.

```
NEW
200  GR
210  REM FARBE WÄHLEN
220  COLOR = INT (16 * RND(1))
230  REM ZUFALLSPUNKT WÄHLEN
240  X = INT (40 * RND(1))
250  Y = INT (40 * RND(1))
260  REM PUNKT ZEICHNEN
270  PLOT X,Y
280  REM WIEDERHOLEN
290  GOTO 220
```

Versuchen Sie, für RND in anderen Programmen Verwendung zu finden. Wie wäre es mit einem Programm, welches Linien in Zufallsfarben quer über den Bildschirm zeichnet?

Unterprogramme

Lassen Sie uns annehmen, Sie hätten ein Spiel, wofür man einen Spielstein braucht. Dieser Stein sollte wie ein blaues Pferd mit weißem Kopf und orangenen Hufen aussehen. Das folgende Programm zeichnet Ihnen den gewünschten Spielstein.

```
NEW
1000  REM PROGRAMM ZUM ZEICHNEN EINES BLAUEN PFERDES MIT WEIS-
      SEM KOPF UND ORANGENEN HUFEN
1010  GR
1020  COLOR = 7 : REM HELLBLAU
1030  PLOT 15,15
1040  HLIN 15,17 AT 16
1050  COLOR = 9 : REM ORANGE
1060  PLOT 15,17
1070  PLOT 17,17
1080  COLOR = 15 : REM WEISS
1090  PLOT 14,15
```

In der Tat, an diesem Programm ist nichts verkehrt. Es zeichnet wirklich ein blaues Pferd mit weißem Kopf und orangenen Hufen. Aber nehmen wir doch einmal an, Sie müßten jetzt noch ein Pferd woanders auf dem Bildschirm zeichnen. Sie könnten dann dieses Programm so umschreiben, daß Sie neue Koordinatenwerte haben, — aber das wird auf die Dauer lästig werden. Es müßte doch eine andere Möglichkeit geben, mit Hilfe desselben Programms eine Figur an beliebiger Stelle auf dem Bildschirm darzustellen, ohne es jedesmal neu zu schreiben.

Mit der folgenden Beobachtung schaffen wir es: Durch Addition eines Wertes zu der Koordinate A des Punkts (A,B) wird dieser Punkt nach rechts verschoben. Der zu addierende Wert entspricht der Verschiebungsdistanz. Der Punkt (4,17) rückt zum Beispiel 10 Spalten nach rechts, wenn wir 10 zu der 4 addieren, d.h. den Punkt (14,17) errechnen.

Auf ähnliche Weise kann der Punkt nach links gerückt werden. Dazu müssen wir eine Zahl abziehen (die Mathematiker reden von der Addition einer negativen Größe). Ein einfaches Experiment wird Sie davon überzeugen, daß sich durch Addieren und Subtrahieren von der zweiten Koordinate der Punkt nach unten oder nach oben bewegt.

Wenn wir uns das merken, können wir das obige Programm so umschreiben, daß der „Mittelpunkt" des Pferdes auf fast jeden Punkt (X,Y) gelegt werden kann. Was heißt hier „fast" jeder Punkt? Wenn wird den Mittelpunkt auf die Bildschirmkante legten, wäre ja doch das Pferd teilweise außerhalb des Bildschirms. Das könnte uns eine Fehlermeldung einbringen, etwa ?ILLEGAL QUANTITY ERROR IN 1030 oder sonstwo im Programm. Hier haben wir also das umgeänderte Programm

```
NEW
1000  REM EIN PFERD AN IRGENDEINER STELLE ZEICHNEN
1010  COLOR = 7 : REM HELLBLAU
1020  PLOT X,Y - 1
1030  HLIN X,X + 2 AT Y
1040  COLOR = 9 : REM ORANGE
1050  PLOT X,Y + 1
1060  PLOT X + 2, Y + 1
1070  COLOR = 15 : REM WEISS
1080  PLOT X - 1,Y-1
```

Sie bemerken schon, wir haben GR weggelassen. Wir wollen ja dieses Programm dazu benutzen, um mehrere Pferde auf den Bildschirm zu setzen. Wenn also GR vorkäme, dann würde jedesmal der Bildschirm beim Zeichnen des nächsten Pferdes gelöscht.

Man kann dieses Programm nicht gleich so, wie es ist, laufen lassen. Zunächst müssen wir GRafikmodus setzen und Werte für X und Y wählen. Als guten ersten Versuch schreiben wir :

```
20  GR
30  REM MITTELPUNKT DES ERSTEN PFERDS
40  X = 12
50  Y = 35
```

Wenn Sie nun das Programm so laufen lassen, erhalten sie zwar ein Pferd an der gewünschten Stelle, aber das Programm wird gleich danach aufhören. Wenn wir nun schreiben würden

```
 60
 70  REM MITTELPUNKT DES ZWEITEN PFERDS
 80  X = 33
 90  Y = 2
100
```

und wenn dann das Programm mit Zeile 1000 noch einmal ausgeführt würde, hätten wir eine einfache und bequeme Lösung, nicht wahr?

Wir wissen schon, der Computer kann mit den merkwürdigen Zeilen 60 und 100 nichts anfangen, aber er kann schon etwas mit

```
GOSUB 1000
```

anfangen, jedenfalls im Applesoft. Wir nennen in diesem Fall die Zeilen 1000 bis 1080 ein **Unterprogramm.** GOSUB befiehlt dem Computer, zum Unterprogramm zu gehen, in diesem Falle also zur Zeile 1000, und dort weiterzumachen. GOSUB sagt dem Computer außerdem, daß es nach Ausführung des Unterprogramms mit der dem GOSUB folgenden Zeile weitergehen soll. Die Beendigung des Unterprogramms wird an dem RETURN -Befehl erkannt. Um also das Pferdezeichnen zu einem Unterprogramm zu vervollständigen, fügen Sie hinzu

```
1090  RETURN
```

Nun können wir also unser Programm ganz unseren Wünschen entsprechend hinschreiben :

```
 20  GR
 30  REM MITTELPUNKT DES ERSTEN PFERDS
 40  X = 12
 50  Y = 35
 60  GOSUB 1000
 70  REM MITTELPUNKT DES ZWEITEN PFERDS
 80  X = 33
 90  Y = 2
100  GOSUB 1000
```

Lassen Sie das Programm jetzt laufen. Sie werden eine Fehlermeldung bekommen :

```
?RETURN WITHOUT GOSUB IN 1090
```

Sonst scheint dem Programm nichts zu fehlen. Wir fassen zusammen : Wir haben einen neuen Befehl konstruiert, und zwar einen Befehl zum Pferdezeichnen. Diesen Befehl können wir vermittels

```
GOSUB 1000
```

anwenden, um unsere besonderen Pferde auf dem Bildschirm an beliebig gewählter Stelle (X,Y) zu zeichnen.

Erfassung eines Ablaufes

Der Programmabschnitt von Zeile 1000 bis 1090 ist, wie gesagt, ein Unterprogramm. Der Abschnitt 20 bis 200 wird **Hauptprogramm** genannt.

Um im einzelnen zu verfolgen, wie die Programmschritte ablaufen, gibt es den besonderen Befehl TRACE. Mit diesem Befehl können Sie herausfinden, weshalb Ihnen der Apple vorhin, als wir das Programm zum Pferdezeichnen laufen ließen, eine Fehlermeldung gab. Fügen Sie dem Hauptprogramm die Zeile

```
10  TRACE
```

hinzu und löschen Sie zunächst die Zeile 20. Versetzen Sie den Apple in Textmodus und lassen Sie das Ganze laufen.

Auf dem Bildschirm erscheinen jetzt die Zeilennummern derjenigen Befehle, die gerade ausgeführt werden. Wie Sie sehen, beginnt die Ausführung mit der Zeile 10 und geht im Hauptprogramm weiter bis zum Aufruf des Unterprogrammes (GOSUB). Danach geht es zurück ins Hauptprogramm, das jetzt wieder weiter ausgeführt wird bis zum nächsten Aufruf des Unterprogramms. Bei der Rückkehr vom Unterprogramm ist nun Zeile 1000 die nächstfolgende Zeile, also geht es dort weiter. Das Unterprogramm wird jetzt zum dritten Mal ausgeführt. Hier haben wir also das Problem. Sie werden jetzt die Fehlermeldung verstehen.

Wir können das Problem beheben, indem wir dem Programm

```
110  END
```

hinzufügen. Wenn die Ausführung zu dieser Zeile gelangt, wird genau das passieren, was das Wort besagt : Die Ausführung wird beendet. Lassen Sie also diese Version laufen. Wie Sie sehen, es gibt keine Fehlermeldung mehr. Wenn Sie nun bloß einen Teil des Programms in der Ausführung verfolgen wollen, dann können Sie das in Verbindung mit dem NOTRACE Befehl erreichen. Fügen Sie die Zeile

```
65 NOTRACE
```

hinzu. Das Programm wird jetzt nur bis zur Ausführung der Zeile 65 mitverfolgt.

Man kann TRACE auch unmittelbar ausführen lassen. Sagen Sie dazu

```
TRACE
RUN
```

dann wird Ihr Programm in der Ausführung aufgezeichnet.



Sobald Sie TRACE befohlen haben, wird die Ausführung des Programms von der Stelle ab aufgezeichnet, auch bei späteren RUN Befehlen. Das geschieht sowohl bei unmittelbarer Ausführung, als auch falls TRACE im Programm steht. Um TRACE abzustellen, müssen Sie NOTRACE befehlen. Das kann als unmittelbarer Befehl oder als Programmzeile, die später ausgeführt wird, geschehen.

Ein Unterprogramm, das Pferde besser zeichnen kann

Unterprogramme sollten derart geschrieben werden, daß bei der Ausführung Fehler möglichst nicht auftreten können. Unser Unterprogramm hat zur Zeit den Nachteil, daß das Pferd für manche X,Y Werte teilweise über die Bildschirmkante herausragen würde. Wir können das folgendermaßen vermeiden :

```
1012  IF X < 1 THEN X = 1
1014  IF X > 37 THEN X = 37
1016  IF Y < 1 THEN Y = 1
1018  IF Y > 38 THEN Y = 38
```

Wissen Sie, warum Y maximal 38 sein kann, während X maximal nur 37 sein darf?

Wenn in dieser Version versucht wird, das Pferd zu nahe an der Bildschirmkante zu zeichnen, wird das Pferd die notwendige Distanz ins Innere des Bildschirms gerückt. Es gäbe auch andere Möglichkeiten, das oben genannte Problem zu korrigieren. Wir könnten zum Beispiel selbst eine Fehlermeldung schreiben und dann das Programm anhalten. Die gerade detaillierte Lösung hat aber den Vorteil, daß das Programm gerade nicht aufhört und man sehen kann, daß etwas geschieht.

Manchmal ist es auch der Fall, daß Sie gewisse Werte im Unterprogramm für unterschiedliche GOSUB Aufrufe ändern möchten. Wenn etwa ein zweiter Spieler ein Pferd mit unterschiedlicher Farbe setzen möchte, können Sie natürlich zu diesem Zweck ein völlig neues Unterprogramm schreiben, aber warum so umständlich? Benutzen wir doch lieber einfach Variable. Anstatt in Zeile 1010 also COLOR = 7 zu sagen, könnten wir ja

```
1010  COLOR = BODY
```

sagen, und ebenso

```
1040 COLOR = FEET
1050 COLOR = FACE
```

Dann könnte man das Hauptprogramm folgendermaßen haben :

```
 20 GR
 30 REM PFERDEFARBEN DES ERSTEN SPIELERS
 40 BODY = 7 : REM HELLBLAU
 50 FEET = 9 : REM ORANGE
 60 FACE = 15 : REM WEISS
 70 REM POSITION DES PFERDES DES ERSTEN SPIELERS
 80 X = 15
 90 Y = 30
100 GOSUB 1000
```

(Versichern Sie sich, daß Sie das END zum Schluß nicht vergessen!) In diesem Programm haben wir schon eine ganze Menge zusätzlicher Befehle außer denen zum Pferdezeichnen. Wir können uns die Sache jedoch noch mehr erleichtern, indem wir ein weiteres Unterprogramm nur zur Wahl der Pferdefarbe schreiben. Dieses neue Unterprogramm ruft dann selbstständig das Zeichnungsunterprogramm auf.

```
2000 REM BLAUES PFERD MIT WEISSEM KOPF UND ORANGENEN HUFEN
2010 BODY = 7 : REM HELLBLAU
2020 FEET = 9 : REM ORANGE
2030 FACE = 15 : REM WEISS
2040 GOSUB 1000
2050 RETURN

2500 REM ORANGENES PFERD MIT ROSA HUFEN UND GRÜNEM KOPF
2510 BODY = 9 : REM ORANGE
2520 FEET = 11 : REM ROSA
2530 FACE = 12 : REM GRÜN
2540 GOSUB 1000
2550 RETURN
```

Wenn Sie jetzt ein blaues Pferd mit einem weißen Kopf und orangenen Hufen auf Position (10,11) zeichnen möchten, vereinfacht es sich auf die folgenden Zeilen :

```
30 REM PFERD DES ERSTEN SPIELERS
40 X = 10
50 Y = 11
60 GOSUB 2000
```

Um zusätzlich das orangene Pferd in Position (19,2) zu haben, brauchen wir lediglich :

```
 70 REM PFERD DES ZWEITEN SPIELERS
 80 X = 19
 90 Y = 2
100 GOSUB 2500
```

Die in Zeilen 2000 bzw. 2800 anfangenden Unterprogramme rufen beide das Unterprogramm in der Zeile 1000 auf. Nun haben wir alles ziemlich rationell organisiert. Wir verschaffen uns zunächst ein Unterprogramm, in dem mögliche Fehler abgeprüft werden. Das Unterprogramm benutzt Variablen, denen im Hauptprogramm (oder in einem weiteren Unterprogramm) Werte zugewiesen werden. Nun können Sie Pyramiden von sich untereinander aufrufenden Unterprogrammen aufbauen. Mit dieser Art Arbeitsteilung wird das Hauptprogramm einfach zu programmieren sein. Kurz, mit diesen drei Unterprogrammen können Sie eine attraktive Pferdeschau auf dem Bildschirm veranstalten.

Zuerst aber noch ein nützliches Unterprogramm :

```
3000  REM ZUFALLSPUNKT WÄHLEN
3010  X = INT (RND(1) * 37) + 1
3020  Y = INT (RND(1) * 38) + 1
3030  RETURN
```

Und jetzt zum Hauptprogramm

```
 10  REM GRAFIKMODUS SETZEN
 20  GR
 30  REM ZUFALLSPUNKT WÄHLEN
 40  GOSUB 3000
 50  REM EIN BLAUES PFERD DARAUF SETZEN
 60  GOSUB 2000
 70  REM ERNEUT ZUFALLSPUNKT WÄHLEN
 80  GOSUB 3000
 90  REM EIN ORANGENES PFERD DARAUF SETZEN
100  GOSUB 2500
110  REM ALLES WIEDERHOLEN
120  GOTO 30
```

So sollte ein Hauptprogramm eines guten Programmierers aussehen : im wesentlichen GOSUB und REM-Befehle. Relativ kurze Unterprogramme sollten die Arbeit erledigen, ein jedes davon sollte einfach zu programmieren sein und, für sich genommen, vollständig sein. Benutzen Sie TRACE, um genau zu verfolgen, wie die Ausführung insgesamt abläuft.

Grafiken mit hoher Auflösung

Wir haben bisher mit Grafiken mit **niedriger Auflösung** gearbeitet. In diesem Abschnitt werden wir lernen, wie man eine andere Art von Grafik erhält, die Grafik mit **hoher Auflösung** (high-resolution graphics). So können Sie erheblich mehr Detail einzeichnen, als mit dem 40 mal 40 Raster. Der neue Raster hat 280 mal 160 Punkte. Die waagerechten Koordinaten werden von 0 links bis 279 rechts gezählt und die senkrechten Koordinaten von 0 oben bis 159 unten.

Hohe Auflösung ist nun nicht schwer zu verstehen, denn sehr oft sind die Befehle genauso, wie die für niedrige Auflösung, außer einem vorgesetzten H (für Hohe Auflösung). Wenn Sie also Grafikmodus mit niedriger Auflösung gut verstehen, wird es Ihnen hier nützen.

Schreiben Sie also

```
HGR
```

Dadurch versetzen Sie sich in Grafikmodus mit hoher Auflösung. Der Befehl macht den Bildschirm zunächst schwarz, außer den vier Zeilen unten, die für anfallenden Text gedacht sind. Wie bei niedriger Auflösung ist es auch bei hoher Auflösung erlaubt, in diesen Textbereich hineinzuzeichnen. Der maximal zulässige Wert für Y ist 192. Allerdings werden Punkte im Textbereich nicht auf dem Bildschirm gezeigt. Wenn der Cursor unsichtbar sein sollte, drücken Sie mehrmals die **RETURN** -Taste, bis er unten auf dem Schirm erscheint.



Der HGR-Befehl kann nicht in der Band- oder Diskettenversion des Applesoft verwendet werden. Diese Art Grafik kommt für Sie nur dann in Frage, wenn Sie wenigstens 24K Speichergröße haben. Wenn das für Sie zutrifft, wird Ihnen Anhang D erklären, wie Sie sich in diesen Modus versetzen können.

Grafik mit hoher Auflösung ist wirklich eine herrliche Sache, aber Sie können es nicht ohne Opfer haben. Sie haben nämlich weniger Farben zur Verfügung. Und zwar gehen die Farben von Ø (schwarz) bis 7 (weiß). Die Farben sind folgendermaßen kodiert :

Ø	Schwarz Nr. 1	4	Schwarz Nr. 2
1	Grün	5	Orange
2	Violett	6	Blau
3	Weiß Nr. 1	7	Weiß Nr. 2

Der genaue Farbton wird von Ihrem Fernsehgerät abhängen. Außerdem spielt es eine Rolle, an welcher Stelle des Bildschirms die Farben gemalt werden. Ein Punkt der Farbe 3, der im Hochauflösungsmodus gezeichnet wird, wird zum Beispiel blau erscheinen, wenn die horizontale Koordinate geradzahlig ist, grün wenn sie ungerade ist, und weiß nur, wenn sowohl die geradzahlige wie auch die ungeradzahlige horizontale Koordinate gezeichnet wird. Das liegt an der Konstruktion Ihres Fernsehgeräts.

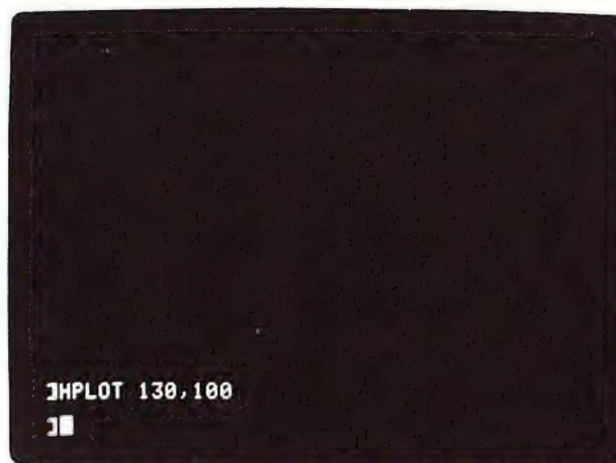


Wenn Sie ein älteres Apple-Modell haben, d.h. mit einer Seriennummer unter 6ØØØ, dann werden die Farben in der ersten Spalte so wie die Farben in der zweiten Spalte aussehen.

Zur Erstellung von Grafiken mit hoher Auflösung gibt es den HPLOT-Befehl. Nachdem Sie HGR befohlen haben, können Sie diesen Befehl mit den Zeilen

```
HCOLOR = 3
HPLOT 13Ø, 1ØØ
```

ausprobieren. Damit wird ein weißer Punkt an der Stelle X = 13Ø, Y = 1ØØ gezeichnet.



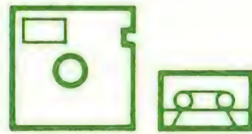
Linien zu zeichnen ist bei hoher Auflösung sogar einfacher als bei niedriger, denn Sie dürfen den Anfangspunkt und den Endpunkt beliebig angeben. Um eine Linie an der oberen Bildschirmkante zu zeichnen, sagen Sie

```
HPLOT Ø,Ø TO 279,Ø
```

Wenn Sie jetzt von der Ecke (279,Ø) aus gerne eine Linie zur anderen Ecke gezeichnet hätten, sagen Sie nun einfach

```
HPLOT TO 279,159
```

und sogleich erscheint eine Linie an der rechten Bildschirmkante. Wenn Sie den HPLOT-Befehl so verwenden, wird als Anfangspunkt der letzte Endpunkt gewählt, und die Linie ist auch in der Farbe des letztgezeichneten Punkts. Das gilt auch, wenn Sie inzwischen mit HCOLOR die Farbe verändert haben. Man kann in einem einzigen Befehl mit dem TO Ketten bilden und mehrere Linien auf einmal zeichnen.

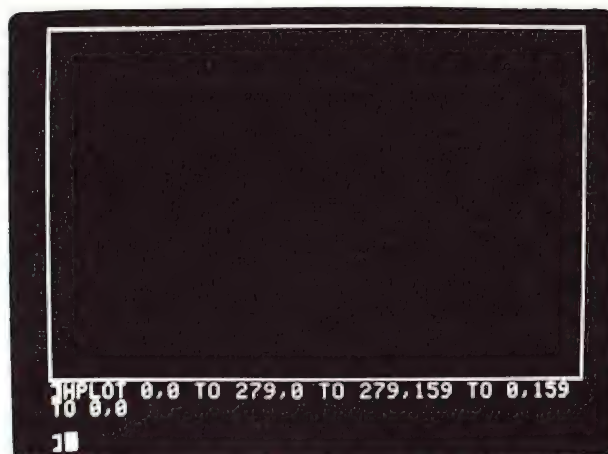


In der Disketten- und Bandversion des Applesoft sind TO-Ketten im HPLOT-Befehl nicht erlaubt. In diesen Versionen dürfen Sie im HPLOT-Befehl also höchstens zwei Punkte angeben, um eine Linie zu zeichnen.

Löschen Sie nun den Bildschirm mit HGR und versuchen Sie folgendes :

```
HPLOT Ø;Ø TO 279,ØTO 279, 159 TO Ø,159 TO Ø,Ø
```

Nun sollte der Bildschirm eingerahmt sein. Wenn das nicht der Fall sein sollte, schauen Sie bitte nach, ob Sie sich nicht vertippt haben. Wenn das der Fall sein sollte, und die Linien sind trotzdem nicht da, dann ändern Sie bitte die Farbe mittels HCOLOR und versuchen es nochmal. Auf manchen Fernsehschirmen erscheinen an manchen Stellen des Bildschirms nur gewisse Farben.



Diagonale Linien sind ebenso einfach zu haben. Eine Linie von links oben nach rechts unten erhält man so :

```
HPLOT Ø,Ø TO 279,159
```

Üben Sie das Linienzeichnen mit unterschiedlichen Längen und Farben.

Das folgende Programm verwandelt den Bildschirm in einen Zeichenblock mit hoher Auflösung.

```
NEW
200  HGR
210  HCOLOR = 3
220  X = PDL(Ø)
230  Y = PDL(1)
240  IF Y > 159 THEN Y = 159
250  HPLOT X,Y
260  GOTO 220
```

Die Zeile 240 muß da sein, da die PDL Funktion Werte bis zu 255 liefern kann. Die Y-Koordinate des Bildschirms geht aber nur bis 159. Lassen Sie nun dieses Programm laufen.

Dieses Programm funktioniert zwar, es wäre aber besser, wenn durchgehende Linien einfacher zu zeichnen wären. Die Lücken, die zwischen den einzelnen Punkten auftreten können, sind nicht immer erwünscht. Sie können das so verbessern :

```
 220  GOSUB 1000
 230  HPLOT X,Y
 240  GOSUB 1000
 250  HPLOT TO X,Y
 260  GOTO 240
1000  X = PDL(0) / .913
1010  Y = PDL(1) / 1.6
1020  RETURN
```

Listen Sie dieses Programm auf und vergewissern Sie sich, daß Sie alles richtig geschrieben haben. Wir erklären nun, was das Programm macht. Wir sind schon im Modus der Grafik mit hoher Auflösung und haben HCOLOR gewählt. Nehmen wir an, das Unterprogramm beginnt nun mit der Zeile 1000. Die X- und Y-Koordinaten werden dadurch festgelegt. Da die Spielkontrollen einen maximalen Wert von 255 liefern können, wir aber X-Werte bis 279 verarbeiten können, dividieren wir den Wert von PDL(0) durch 255/279 = .913. Damit erreichen wir alle X-Werte. Ferner dividieren wir PDL(1) durch 255/159 = 1.6, damit die Y-Koordinate zwischen 0 und 159 liegt. Wie im Fall der niedrigen Auflösung, werden die Endwerte auf ganze Zahlen abgerundet. Via **RETURN** kehren wir zum Hauptprogramm zurück und machen mit der Zeile 230 weiter, zeichnen also diesen Punkt. Danach wird das Unterprogramm erneut aufgerufen und liefert uns einen neuen Punkt. In Zeile 250, also nach **RETURN** , wird jetzt eine Linie vom alten zum neuen Punkt gezeichnet. Dieser Vorgang, außer dem Zeichnen des ersten Punkts, des HGR und des HCOLOR-Befehls, wird dann mit dem GOTO in Zeile 260 wiederholt. Sie benutzen die **CTRL-C** Tastenkombination, um das Programm anzuhalten.

Aus gutem Grund zeichnen wir Linien statt Punkte : Es dauert seine Zeit, bis der Apple einen Punkt auf den Bildschirm gebracht hat. Wenn nun der Apple die Punkte einzeln zeichnen müßte, kommt er nicht mehr mit den Spielkontrollen mit. Daher gab es also die Zwischenräume zwischen den Punkten in der vorigen Version des Programms, wenn an den Spielkontrollen zu schnell gedreht wurde. Wir haben das dadurch behoben, daß wir kurze Linien vom letzten Punkt bis zum neuen Punkt zeichneten, je nach Stellung der Spielkontrollen. Auf diese Weise können die einzelnen Punkte, die auf den Linien liegen, bedeutend schneller gezeichnet werden.

Speichern Sie nun das Programm mittels SAVE, nun es für später zu haben. Lassen Sie es erst danach laufen.

Hier noch ein Programm, das hübsche Moirémuster auf Ihrem Bildschirm zeichnet.

```
NEW
 90  HOME
100  VTAB 24 : REM CURSOR AUF DIE LETZTE ZEILE BRINGEN
120  HGR : REM GRAFIK MIT HOHER AUFLÖSUNG
140  A = RND(1) * 279 : REM ZENTRUM ,,A" WÄHLEN
160  B = RND(1) * 159 : REM ZENTRUM ,,B" WÄHLEN
180  N = INT (RND(1) * 4) + 2 : REM SCHRITTWEITE WÄHLEN
200  HTAB 15 : PRINT ,,SCHRITTWEITE"; N;
220  FOR X = 0 TO 278 STEP N : REM A WERTE DURCHSCHREITEN
240  FOR S = 0 TO 1 : REM 2 LINIEN, VON X UND X+1 AUS
260  HCOLOR = 7 * S : REM ERSTE LINIE SCHWARZ, ZWEITE WEISS
280  REM LINIE DURCH DAS ZENTRUM ZUR ANDEREN SEITE HIN ZEICHNEN
300  HPLOT X + S,0 TO A,B TO 279 – X – S,159
```

```
320  NEXT S,X
340  FOR Y = 0 TO 158 STEP N : REM B WERTE DURCHSCHREITEN
360  FOR S = 0 TO 1 : REM 2 LINIEN, VON B UND B+1
380  HCOLOR = 7 * S : REM ERSTE LINIE SCHWARZ, ZWEITE WEISS
400  REM LINIE DURCH DAS ZENTRUM ZUR ANDEREN SEITE HIN ZEICHNEN
420  HPLOT 279,Y + S TO A,B TO 0,159 - Y - S
440  NEXT S,Y
460  FOR PAUSE = 1 TO 1500 : NEXT PAUSE : REM ETWAS WARTEN
480  GOTO 120
```

Dieses Programm ist schon etwas länger. Geben Sie es sorgfältig ein, und listen Sie es in Segmenten auf (etwa LIST 0,320). Vergewissern Sie sich, daß Ihnen keine Fehler unterlaufen sind. Wenn Sie sicher sind, daß alles stimmt, lassen Sie das Programm laufen.



Wenn Sie die Band- oder Diskettenversion des Applesoft haben, müssen Sie die PLOT- Ketten in den Zeilen 300 und 420 mit separaten HPLOT-Befehlen machen. Also etwa :

```
300  HPLOT X + S,0 TO A,B
310  HPLOT TO 279 - X -S,159

420  HPLOT 279,Y + S TO A,B
430  HPLOT TO 0,159 - Y - S
```

Wie Sie an Zeilen 320 und 440 sehen, kann mit einem NEXT der Abschluß von zwei FOR-Schleifen erreicht werden. Passen Sie auf, daß dabei die Variablennamen in der richtigen Reihenfolge stehen, so daß eine teilweise Überschneidung der Schleifen vermieden wird.

Nun, zurück zur Programmierung. Halten Sie das Programm mit



an und schreiben Sie

```
TEXT
```

Haben Sie gute Ideen, wie das Programm verändert werden kann? Nachdem Sie sich diese Version mit SAVE auf Platte oder Band gespeichert haben, können Sie zum Beispiel die Farbwahl in HCOLOR dem Zufall überlassen. Versuchen Sie auch, abwechselnd blaue und orangene Linien zu zeichnen oder gar nur blaue Linien.

Es gäbe eine Menge mehr über hohe Auflösungsgrafik zu sagen, als wir das hier getan haben. Wenn Sie die gerade besprochenen Befehle sicher beherrschen, können Sie in Kapitel 8 und 9 des Applesoft BASIC-Programmierhandbuchs mehr über die Fähigkeiten Ihres Apples in diesem Modus erfahren.

KAPITEL 5
Zeichenketten und Tabellen

Zeichenketten schleppen

Hätten Sie gerne Ihren Namen rückwärts geschrieben gesehen? Bislang haben wir mit Zeichnungen und Zahlen gespielt, aber ein Computer kann auch Buchstaben und Symbole manipulieren. Er kann dabei mit einzelnen Buchstaben arbeiten oder mit ganzen Zeichenketten auf einmal. Das wird Ihnen ziemlich normal erscheinen, denn Sie machen es ja ebenso im täglichen Leben. Variablen, die Zeichenketten als Werte annehmen, haben ebenso Namen wie Variablen, die numerische Werte haben. Im Falle von Zeichenkettenvariablen (auf englisch "Strings"), wird allerdings zum abschluß ein Dollarsymbol geschrieben ($). Wir zeigen Ihnen einige Namen als Beispiel :

```
MYNAME$
A$
SENTENCES$
```

Die Variable A wird von der Variablen A$ unterschieden. Man darf beide im gleichen Programm verwenden.

Wenn Sie nun der Variablen NAME$ (Sie können "Name-Dollar" sagen) den Wert ,,HARRY S. TRUMAN" zuweisen möchten, können Sie schreiben

```
NAME$ = "HARRY S. TRUMAN"
```

Beachten Sie, daß die Zeichenkette in Apostrophe eingeschlossen wird. Der Befehl

```
PRINT NAME$
```

wird den Wert der Variablen NAME$ drucken, in diesem Falle also den Namen des 33. Präsidenten der Vereinigten Staaten. Wenn Sie nun eine Zeichenkette haben, die Sie oft benötigen, können Sie sie als Wert einer Variablen mit einem kurzen Namen zuweisen.

Mehrere Applesoft-Befehle manipulieren Zeichenketten. Nehmen wir an, Sie wollten die länge einer Zeichenkette wissen, d.h., wieviele Zeichen sie enthält. Dann schreiben Sie

```
PRINT LEN ("HARRY S. TRUMAN")
```

oder auch

```
PRINT LEN (NAME$)
```

Daraufhin wird der Apple 15 als länge dieser Zeichenkette angeben. Beachten Sie, daß die Leerzeichen mitgezählt werden.

Die Länge einer Zeichenkette darf zwischen Ø und 255 liegen. Wenn Sie mehr als 255 Zeichen in eine Kette setzen, werden Sie die Fehlermeldung ?SYNTAX ERROR oder STRING TOO LONG ERROR erhalten. Die Zeichenkette der Länge null heißt **leere** Zeichenkette. Im Applesolft BASIC Programmierhandbuch können Sie weiteres über die leere Zeichenkette erfahren.

Manchmal möchten Sie nur Teile einer Zeichenkette drucken. Dafür gibt es die hübschen Funktionen LEFT$, MID$ und RIGHT$.

Wenn Sie zum Beispiel die ersten fünf Zeichen in NAME$ drucken wollen, können Sie

```
PRINT LEFT$ (NAME$, 5)
```

schreiben. Gleich wird

HARRY

auf dem Bildschirm erscheinen. Und wenn Sie

PRINT RIGHT$ (NAME$, 5)

schreiben, so wird

RUMAN

erscheinen. In jedem Programm, das Sie unter Verwendung von Zeichenkettenvariablen schreiben, müssen Sie diesen Variablen **innerhalb** des Programms Werte zuweisen. Wenn Sie deb RUN Befehl geben, erhalten jedesmal zunächst alle numerischen Variablen den Wert Ø und alle Zeichenkettenvariablen die leere Zeichenkette als Wert. Wir haben hier gleich ein kurzes Programm, in dem die Funktionen LEN und LEFT$ benutzt werden.

```
NEW
 90  NAME$ = "HARRY S. TRUMAN"
100  FOR N = 1 TO LEN (NAME$)
110  PRINT LEFT$ (NAME$, N)
120  NEXT N
```

Lassen Sie dieses Programm laufen. Die RIGHT$-Funktion ist geradeso wie die LEFT$-Funktion, nur werden die Zeichen vom rechten Ende genommen. Modifizieren Sie nun das Programm und lassen Sie es mit RIGHT$ statt LEFT$ laufen. Was passiert dabei?

Wenn Sie gerne Zeichen aus der Mitte, statt am Anfang oder am Ende einer Zeichenkette herausholen wollen, müssen Sie die MID$ Funktion verwenden.

Schreiben Sie

PRINT MID$ (NAME$, 7)

Ihr Computer antwortet

S. TRUMAN

Denn S ist das siebente Zeichen in der Kette. Versuchen Sie nun

```
NEW
190 NAME$ = "HARRY S. TRUMAN"
200  FOR N = 1 TO LEN (NAME$)
210  PRINT MID$ (NAME$, N)
220  NEXT N
```

Geht alles wie erwartet, wenn Sie das Programm laufen lassen?

Nehmen wir nun an, Sie wollten nur die Buchstaben „Y S. T" in der Zeichenkette in NAME$ haben. Das erreichen Sie, indem Sie ein weiteres Argument zur Funktion MID$ hinzufügen.

PRINT MID$ (NAME$, 5, 6)

Die erste Zahl (5) gibt die Stelle in der Kette an, an der Sie mit dem Schreiben anfangen wollen. Die zweite Zahl (6) besagt, wieviele Zeichen es sein sollen. Daher besagt der Befehl insgesamt: „Finde den fünften Buchstaben der Kette in NAME$ und drucke sechs Zeichen von dem gefundenen fünften aus nach rechts". Ändern Sie also Zeile 21Ø des obigen Programms um in:

```
210  PRINT MID$ (NAME$, N, 6)
```

Machen Sie jetzt im Buch nicht weiter, bis Sie die Funktionen LEFT$, MID$ und RIGHT$ gründlich ausprobiert haben. Sie können sich auch das folgende Programm gründlich überlegen :

```
NEW
200  A$ = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
210  PRINT
220  PRINT "SCHREIBEN SIE EINE ZAHL ZWISCHEN 1 UND "; LEN (A$);
230  PRINT "DANN WIRD IHNEN GESAGT, DER WIEVIELTE BUCHSTABE DAS
     IST";
240  INPUT P
250  IF P > LEN (A$) OR P < 1 THEN GOTO 210
260  PRINT
260  PRINT
270  PRINT MID$ (A$, P, 1); " IST DER BUCHSTABE Nr. "P;" IM ALPHABET
280  PRINT : PRINT
290  PRINT "SCHREIBEN SIE EINEN BUCHSTABEN. DANN WERDE ICH INHEN
     SAGEN"
300  INPUT "DER WIEVIELTE BUCHSTABE IM ALPHABET ER IST"; X$
310  FOR N = 1 TO LEN (A$)
320  IF MID$ (A$, N, 1) = X$ THEN GOTO 370
330  NEXT N
340  PRINT
350  PRINT "DAS IST ABER KEIN BUCHSTABE." : PRINT
360  GOTO 290
370  PRINT
380  PRINT X$;" IST BUCHSTABE NR.   ";N;"   DES ALPHABETS."
390  PRINT
400  GOTO 210
```

Dieses Programm illustriert übliche Programmierpraktiken. Beachten Sie, wie der Buchstabe im Alphabet gefunden wird. Häufig wird diese Art Schleife benutzt, um sich durch eine Zeichenkette zu arbeiten. Beachten Sie auch, wo in den zu druckenden Zeichenketten Leerzeichen stehen. Was würde geschehen, wenn man sie nicht hätte? Schließlich beachten Sie noch, wie die Programmschleife durch LEN (A$) als Grenzwert begrenzt wird, nicht durch eine Zahl wie etwa 26. Das erleichtert es Ihnen, später in anderen Alphabeten zu suchen. Versichen Sie es mal, etwa nur mit Konsonanten, und machen Sie sich damit vertraut.

Sie dürfen den Zeichenkettenwert einer Variablen einer anderen zuweisen, etwa

```
X$ = A$
```

Mit diesem Befehl wird der Inhalt von A$ nach X$ kopiert. Allerdings dürfen Sie dabei nicht die Funktionen LEFT$, RIGHT$ und MID$ linkseitig verwenden. Zum Beispiel

```
MID$ (X$, 3, 3) = "XYZ"
```

ist nicht erlaubt, wohingegen

```
X$ = MID$ (A$, 24, 3)
```

durchaus erlaubt ist. Nur Variablen dürfen links vom Zuweisungszeichen (=) stehen.

Ach ja — wollen Sie immer noch Ihren Namen rückwärts geschrieben sehen? Das umseitige Programm wird es für Sie tun.

```
NEW
100  REM PROGRAMM DAS IHREN NAMEN RÜCKWÄRTS BUCHSTABIERT
110  INPUT " SCHREIBEN SIE IHREN NAMEN UND ICH WERDE IHN IHNEN
     RÜCKWÄRTS SCHREIBEN" ;N$
120  REM BUCHSTABENFOLGE UMKEHREN
130  FOR T = LEN (N$) TO 1 STEP - 1
140  R$ = R$ + (MID$(N$, T, 1))
150  NEXT T
160  PRINT : PRINT "IHR NAME LAUTET RÜCKWÄRTS" ;R$
170  PRINT : PRINT
180  GOTO 110
```

Lassen Sie dieses Programm laufen und geben Sie verschiedene Namen ein. Nachdem das Programm eine Weile läuft, werden Sie wahrscheinlich bemerken, daß da etwas falsch ist. Zeile 140 ist der Schlüssel zum Problem. Wenn Ihr Name etwa PETRA ist, würden Sie ihn also eingeben; N$ würde diesen Wert annehmen und R$ damit den Wert ARTEP. Vielleicht ist auch gerade Ihr Freund PETER da, der möchte es auch mal probieren. Bei der nächsten Abfrage lassen Sie ihn seinen Namen schreiben. Somit nimmt N$ jetzt den Wert PETER an. In Zeile 140 erhält nun R$ den alten R$ wert **plus** N$ rückwärts. Mit anderen Worten, es wird ARTEP RETEP daraus. Was hier gebraucht wird, wäre ein Befehl, einer Variablen die leere Zeichenfolge zuzuweisen, so daß anschließend R$ richtig aufgebaut werden kann.

Glücklicherweise gibt es diesen Befehl. Es ist der CLEAR Befehl. CLEAR löscht alle Variablenwerte zu 0. Fügen Sie Ihrem Programm bei

```
175  CLEAR
```

Lassen Sie nun das Programm erneut laufen.

Den Clear-Befehl kann man auch zu sofortiger Ausführung benutzen. Schreiben Sie

```
N = 254
PRINT N
```

und danach

```
CLEAR
```

und wieder

```
PRINT N
```

Da haben Sie doch zuletzt eine Null erhalten, nicht wahr?

Haben Sie sich verknüpft?

Sie dürfen eine zweite Zeichenkette an eine erste anhängen und zwar mit dem Plussymbol (+). Das nennt man **Verknüpfung.** Versuchen Sie dies :

```
C$ = „GUTEN MORGEN"
D$ = C$ + „   " + „BILL"
PRINT D$
```

Ihr Apple wird

GUTEN MORGEN BILL

drucken. Verknüpfung ist besonders dann von Nutzen, wenn Sie eine Zeichenkette zunächst auseinandernehmen und dann die Teile wieder mit kleinen Veränderungen zusammenfügen. Wenn Sie zum Beispiel in D$ die Leerzeichen durch Bindestriche ersetzen möchten, könnten Sie das so erreichen

```
E$ = RIGHT$ (D$, 4) + „–" + LEFT$(D$, 5) + „–" + MID$ (D$, 7, 6)
PRINT E$
```

dann wird

BILL-GUTEN-MORGEN

auf Ihrem Bildschirm erscheinen.

Das folgende Programm benutzt auch die Verknüpfung von Zeichenketten.

```
NEW
100  INPUT "GEBEN SIE MIR DIE ERSTE HÄLFTE EINES SATZES EIN! "; HALF$
110  INPUT "UND NUN DIE ANDERE HÄLFTE! ";OTHERHALF$
120  WHOLE$ = HALF$ + OTHERHALF$
130  PRINT
140  PRINT WHOLE$
150  PRINT : PRINT : PRINT : GOTO 100
```

Nun, so wird das mit der Verknüpfung gemacht.

Weitere Funktionen für Zeichenketten

Zeichenketten darf man mit fast allen Zeichen einschließlich Zahlen formen. Allerdings können Zahlen zwischen Anführungszeichen, wie beim PRINT-Befehl auch, nicht als numerische Werte behandelt werden. Man kann nicht direkt mit ihnen rechnen. Um diesen Sachverhalt zu verfolgen, schreiben Sie

```
C$ = "123"
PRINT C$ + 7
```

Verwirrt wird der Apple schreiben

?TYPE MISMATCH ERROR

und nicht in der Lage sein, mit dem letzten Befehl fertig zu werden. Wir müssen die VAL-Funktion zu Hilfe nehmen :

VAL

(Abkürzung für--Wert)

Die VAL-Funktion gibt Ihnen den numerischen Wert einer Zeichenkette anstatt der Zeichenkette selbst. Schreiben Sie

```
PRINT C$
```

und danach

```
PRINT VAL (C$)
```

Bei beiden Befehlen bekommen Sie die gleiche Antwort. Aber der Schein kann trügen. Sie wissen schon, bei

```
PRINT C$ + 5
```

kommt

```
?TYPE MISMATCH ERROR
```

heraus. Aber versuchen Sie mal

```
PRINT VAL (C$) +
```

und

```
128
```

erscheint auf dem Bildschirm. Beachten Sie, daß der Name der Variable in Klammern eingeschlossen werden muß, da er das Argument der VAL-Funktion darstellt.

Wie können wir nun den Wert von C$ an eine Variable zuweisen, die keine Zeichenkettenvariable ist?
Sehr einfach! Schreiben Sie

```
Q = VAL (C$) – 21
```

Schauen Sie mit

```
PRINT Q
```

nach, was dabei herauskommt. Hat Q den erwarteten Wert? Sie können sogar VAL zum Addieren von Zeichenketten verwenden. Versuchen Sie das so :

```
K$ = "12"
```

gefolgt von

```
P = VAL (C$) + VAL (K$)
PRINT P
```

Probieren Sie VAL noch mit anderen Zeichenketten aus! Versuchen Sie auch Zeichenketten, bei denen am Anfang oder am Ende Buchstaben stehen.

Manchmal ist es auch notwendig, eine Zahl in eine Zeichenkette zu verwandeln. Die STR$-Funktion besorgt das, die ansonsten eine Umkehrung von VAL ist. Nehmen wir also an, Sie wollten den (numerischen) Wert von P in eine Zeichenkette verwandeln und einer anderen Variable zuweisen. Sie werden an

```
P$ = STR$ (P)
PRINT P$
```

sehen, wie es gemacht wird. Hier ist nun ein Programm, das sowohl STR$ als auch VAL verwendet.

```
3ØØ  INPUT "SCHREIBEN SIE EINE ZAHL ZWISCHEN 1 UND 999999999 "; N$
31Ø  N = VAL (N$)
32Ø  IF N < 1 OR N > 999999999 THEN GOTO 3ØØ
33Ø  N$ = STR$ (N)
34Ø  FOR T = LEN (N$) TO 1 STEP - 1
35Ø  P$ = P$ + MID$ (N$, T, 1)
36Ø  NEXT T
37Ø  PRINT : PRINT "ORIGINAL" ,N$
38Ø  PRINT : PRINT "UMGEKEHRT" ,P$
39Ø  P = VAL (P$)
4ØØ  PRINT : PRINT "ORIGINAL + UMGEKEHRT = "; N + P
41Ø  CLEAR
42Ø  PRINT : PRINT : GOTO3ØØ
```

Verstehen Sie das Programm? Warum haben wir Kommata in Zeilen 37Ø und 38Ø? Löschen Sie nun Zeile 33Ø, um die Antwort herauszufinden. Die ersten vier Zeilen des Programms zeigen Ihnen den Weg, ein wahrhaft narrensicheres Eingabe-Teilprogramm zu schreiben. Überprüfen Sie, welche Eingaben dieses Programm noch stoppen könnten. Verändern Sie es dann so, daß Ihre Fehlereingaben abgefangen werden, bevor sie das Programm zum Halten bringen können.

Tabellen vorstellen

In diesem Abschnitt werden wir Beispiele aus der Mathematik benutzen, um Tabellen zu behandeln. Es wird aber Unterhaltungsmathematik sein (– das gibt es!). Außer einem bißchen Kopfrechnen, brauchen Sie sonst keine weiteren Vorkenntnisse.

Mit Tabellen können Sie im Nu auf eine beliebige Eintragung zugreifen. Damit werden Ihnen zusätzliche Möglichkeiten in der Programierung verschafft, die das zusätzliche Nachdenken und Experimentieren, das Sie zunächst hineinstecken müssen, mehr als aufwiegen.

Bei unseren Tabellen handelt es sich um Zahlentabellen. Der **Tabellenname** ist wie jeder andere Variablenname auch : zum Beispiel könnte man A wählen. Der Tabellenname A wird vom Variablennamen A unterschieden.

Um eine Tabelle zu erstellen, müssen Sie zunächst dem Computer sagen, wieviele Eintragungen Sie maximal haben wollen. Sie machen das mit dem DIM-Befehl (DIM kürzt Dimension ab). Die Tabellenelemente werden von Ø ab aufsteigend numeriert. Um also eine Tabelle mit maximal 16 Elementen zu bekommen, schreiben Sie

```
DIM A(15)
```

Dieser Befehl hat uns 16 neue Variablen verschafft. Diese Variablen werden wie die gewöhnlichen Variablen gebraucht, die Sie ja nun schon hinlänglich kennen und vielleicht sogar lieben. Die neuen Variablen sind

```
A (Ø)
A (1)
A (2)
```

u.s.w. bis

```
A (15)
```

Vielleicht kommt Ihnen das etwas umständlich vor, aber jedenfalls werden sie so wie gewöhnliche Variablen gebraucht. Der Befehl

```
A (9) = 45 + A (12)
```

ist völlig richtig. Die Zahl in der Klammer nennt man **Index,** und A (12) können Sie „A bei 12" aussprechen.

Als Index dürfen Sie einen arithmetischen Ausdruck oder eine Variable benutzen.

Schreiben Sie das folgende Programm. Es veranschaulicht den Gebrauch von Variablen als Index und schreibt Ihnen den Inhalt jedes Tabellenelements.

```
100  REM TABELLE "DAYS" AUF 7 ELEMENTE VEREINBAREN
110  DIM DAYS (6)
120  REM DIE TABELLE MIT WERTEN FÜLLEN
130  FOR NUM = Ø TO 6
140  DAYS (NUM) = NUM + 1
150  NEXT NUM
160  REM TABELLENWERTE SCHREIBEN
170  FOR I = Ø TO 6
180  PRINT "DAYS" (";I;") = ";DAYS (I)
190  NEXT I
```

Wenn eine Tabelle benutzt wird, bevor Sie ihre Größe vereinbart haben, dann wird Applesoft Ihnen 11 Plätze zuteilen (von Ø bis 10). Aber es ist gute Form, Tabellen **immer** in Größe zu vereinbaren, und zwar bevor sie benutzt werden.

Nehmen Sie an, Sie wollen ein Programm schreiben, welches Ihnen die Zahlen zwischen 1 und 8 in gemischter Ordnung schreiben soll.
Um das zu erreichen, müssen Sie mit Tabellen jonglieren. Das kann man gerade mit Tabellen ausgezeichnet tun. Das folgende Programm zeigt Ihnen wie :

```
NEW
200  REM TABELLENGRÖSSE VEREINBAREN
210  DIM GLASS (8)
220  REM TABELLE FÜLLEN
230  FOR I = 1 TO 8
240  GLASS (I) = I
250  NEXT I
260  REM TABELLE MISCHEN UND EIN ELEMENT WÄHLEN
270  FOR WINE = 1 TO 8
280  REM ANDERES ELEMENT WÄHLEN
290  MILK = INT (RND (1) * 8) + 1
300  REM SIND MILK UND WINE VERSCHIEDEN?
310  REM WENN NICHT, NOCHMAL VERSUCHEN
320  IF MILK = WINE THEN GOTO 28Ø
330  REM GLASS (WINE) UND GLASS (MILK) VERTAUSCHEN
340  TEMP = GLASS (WINE) : GLASS (WINE) = GLASS (MILK) : GLASS (MILK)
     = TEMP
350  NEXT WINE
360  REM TABELLE AUSSCHREIBEN
370  FOR C = 1 TO 8
380  PRINT GLASS (C)
390  NEXT C
```

Verstehen Sie, wie das Programm funktioniert? Zuerst füllt es eine Tabelle mit Zahlen, dann mischt es die Tabelle. Beachten Sie, daß Sie beim Füllen nicht mit dem Index 0 anfangen mußten. Wir beschreiben nun, was einige dieser esoterischen Programmzeilen im einzelnen tun. Zeilen 230 bis 250 füllen die Tabelle und weisen jedem Element als Wert seinen Index zu. GLASS (1) = 1, usw. Zeile 270 weist der neuen Variablen WINE nacheinander die Werte 1 bis 8 zu. Zeile 280 weist MILK einen Zufallswert zwischen 1 und 8 zu. Zeile 320 bewirkt, daß der Wert von MILK immer von dem Wert von WINE verschieden ist. Daraufhin vertauscht Zeile 340 die Werte von GLASS(WINE) und GLASS(MILK). Schließlich wird das Resultat in den Zeilen 370 bis 390 geschrieben.

Die Vertauschung der Werte in der Zeile 340 kann man sich folgendermaßen veranschaulichen. Nehmen wir an, wir hätten zwei Gläser, eins mit Wein (WINE), das andere mit Milch (MILK). Wir wollen die Milch in das Weinglas schütten und den Wein ins Milchglas. Glücklicherweise haben wir ein drittes Glas (TEMP). Wir schütten also den Wein in das Glas TEMP, dann die Milch in das Weinglas, sodann den Wein aus TEMP ins Milchglas. Damit hätten wir den Inhalt der beiden Gläser vertauscht.

Fehlermeldungen bei Tabellen

Hier sind einige Fehlermeldungen, die Sie beim Arbeiten mit Tabellen erhalten könnten.

?REDIM'D ARRAY

Dieser Fehler tritt auf, wenn Sie für eine Tabelle mehrmals im Programm eine Größe vereinbaren wollen. Häufig passiert dies', wenn Sie auf die Tabelle zugreifen, bevor Sie sie vereinbart haben, denn Sie bekommen dabei ja 11 Elemente. Sie fügen vielleicht später eine Vereinbarung hinzu, die aber vielleicht nicht am Programmanfang liegt.

?BAD SUBSCRIPT ERROR

Sie bekommen diese Nachricht, wenn Sie auf ein Tabellenelement zugreifen wollen, das außerhalb des vereinbarten Bereichs liegt. Wenn Sie zum Beispiel A mit 26 Elementen vereinbaren, also durch DIM A (25), und nun auf Element A (52) zugreifen, oder auf ein anderes Element, dessen Index nicht zwischen 0 und 25 liegt. Sie erhalten dabei ?BAD SUBSCRIPT ERROR

?ILLEGAL QUANTITY ERROR

Sie erhalten diese Nachricht, wenn Sie eine negative Zahl als Index in der Tabelle benutzen.

Wir haben Ihnen hiermit einige der Möglichkeiten, die sich Ihnen mit Tabellen erschließen, vorgestellt. Wir haben nur eindimensionale Tabellen benutzt. Sie dürfen auch mit Tabellen arbeiten, die zwei oder mehr Dimensionen haben. Schauen Sie im Applesoft BASIC-Programmierhandbuch nach, wie man das macht.

Zusammenfassung

In diesem Buch haben wir Ihnen den Kern des Applesoft BASIC vorgestellt. Wenn Sie nun dieses Buch nochmal wiederholen und die ganzen Programme mitschreiben und ausprobieren, werden Sie Ihre neugewonnenen Kenntnisse beträchtlich vertiefen und festigen. Der Apple kann aber noch viel mehr, und nachdem Sie dieses Buch gemeistert haben, können Sie noch viele neue Welten erforschen.

ANHÄNGE

FLASH

Der Videomodus wird auf Blinken geschaltet. Die Ausgabe vom Computer wird auf dem Bildschirm abwechselnd in weißen Zeichen auf schwarzem Hintergrund und schwarzen Zeichen auf weißen Hintergrund gezeigt. Mit dem NORMAL Befehl wird diese Blinken wieder abgestellt, und Sie erhalten wieder weiße Zeichen auf schwarzem Hintergrund.

FOR W = 1 TO 20 . . . NEXT W
FOR Q = 2 TO – 3 STEP – 2 . . . NEXT Q
FOR Z = 5 TO 4 STEP 3 . . . NEXT Z

Hiermit können Sie Schleifen schreiben, in denen die Instruktionen zwischen dem FOR Befehl und dem Next Befehl (sas Oben und Unten der Schleife) in vorgeschriebener Anzahl ausgeführt werden. Im ersten Beispiel zählt W, wie oft die Schleife ausgeführt wird.

W nimmt dabei nacheinander die Werte 1, 2, 3, . . .20 an. Bei Beendigung der Schleife hat W den Wert 21, und der auf NEXT W folgende Befehl wird ausgeführt. Die anderen Beispiele veranschaulichen, wie die Schrittweite (STEP) der Schleife gewählt wird, falls sie nicht 1 sein soll. Die Überprüfung, ob die Schleifenvariable den Grenzwert überschreitet, findet am **Schleifenende** statt, und daher wird die dritte Beispielsschleife einmal ausgeführt.

GOSUB 250

Die Programmausführung wird zur angegebenen Zeile verzweigt (hier Zeile 250). Der nächste RETURN Befehl bewirkt eine Rückverzweigung, bei der die Ausführung mit der Zeile, die auf GOSUB folgt, fortgeführt wird. Dabei handelt es sich um die zuletzt ausgeführte GOSUB Instruktion, zu der noch kein RETURN ausgeführt worden ist.

GOTO 250

Die Programmausführung verzweigt zur angegebenen Zeile (hier also die Zeile 250).

GR

Versetzt den Apple in Grafikmodus mit niedriger Auflösung mit einem Bildschirmraster von 40 mal 40 Rechtecken. Unten im Bild werden 4 Textzeilen reserviert. Der Bildschirm wird auf Schwarz gelöscht, ebenso COLOR, die Zeichenfarbe. Der Cursor wird auf die Textzeilen gesetzt.

HCOLOR = 4

In Grafikmodus mit hoher Auflösung wird die angegebene Farbe zum Zeichnen gewählt. Es gibt die folgenden Kodes :

0 Schwarz 1	4 Schwarz 2
1 Grün	5 Orange
2 Violett	6 Blau
3 Weiß 1	7 Weiß 2

Bei älteren Modellen des Apple (Seriennummern unter 6000) sehen die Farben in der zweiten Spalte ebenso aus wie die in der ersten.

HGR

Dieser Befehl ist nur in der Firmware-Version des Applesoft möglich. Er versetzt den Apple in Grafikmodus mit hoher Auflösung mit einem Bildschirm Raster von 280 mal 160 Rechtecken. An der Unterkante des Bildschirms bleiben vier Textzeilen. Das Bild wird auf Schwarz gelöscht, und Seite 1 des Speichers wird dargestellt. Weder HCOLOR noch der Speicher für die Texteingabe werden verändert. Der Cursor wird nicht auf die nun noch sichtbaren Zeilen gebracht.

HGR2

Grafikmodus mit hoher Auflösung unter Benutzung des gesamten Bildschirms (280 mal 192 Raster). Das Bild wird auf Schwarz gelöscht, und Seite 2 des Speichers dargestellt. Der Speicher für Texteingabe bleibt unberührt.

HLIN 10,20 AT 30

Hiermit werden horizontale Linien im Grafikmodus mit niedriger Auflösung gezeichnet, und zwar in der zuletzt mit COLOR gewählten Farbe. Die Koordinaten des Bilds fangen links oben mit x = 0 und y = 0 an. In unserem Beispiel wird eine Gerade von x = 10 nach x = 20 auf der Höhe y = 30 gezeichnet, d.h. von (10, 30) nach (20, 30).

HOME

Der Cursor wird nach links oben im Textfenster bewegt. Der gesamte Text auf dem Bildschirm wird gelöscht.

```
HPLOT 10, 20
HPLOT 30, 40 TO 50, 60
HPLOT TO 70, 80
```

Punkte oder Linien werden im Grafikmodus mit hoher Auflösung in der zuletzt mit HCOLOR gewählten Farbe gezeichnet. Die Koordinaten fangen links oben mit x = 0, y = 0 an. Im ersten Beispiel wird ein Punkt an der Stelle x = 10, y = 20 gezeichnet. Im zweiten Beispiel wird eine Linie vom Punkt x = 30, Y = 40 zum Punkt x = 50, y = 60 gezeichnet. Im dritten Beispiel wird eine Linie vom zuletzt gezeichneten Punkt (Endpunkt) zum Punkt 70, 80 gezogen, und zwar in derselben Farbe wie der Ausgangspunkt. HCOLOR kann zwischenzeitlich durchaus verändert worden sein.

HTAB 23

Der Cursor wird nach links oder nach rechts in die angegebene Spalte bewegt. Im Beispiel wird der Cursor in die Spalte 23 bewegt

```
IF AGE < 18 THEN A = 0 : B = 1 : C= 2
IF ANS$ = "YES" THEN GOTO 100
IF N < MAX THEN GOTO 25
```

Wenn der Ausdruck zwischen IF und THEN den Wert „richtig" hat (also ein von Null verschiedener Wert), dann werden die Befehle, die dem THEN auf der gleichen Zeile folgen, ausgeführt. Sonst werden diese Befehle ignoriert, und die Ausführung fährt fort mit der dem IF im Programm folgenden Zeile. Zeichenketten werden alphabetisch verglichen.

```
INPUT  A
INPUT "TYPE AGE THEN A COMMA THEN NAME. ";B, C$
```

Im ersten Beispiel wird ein Fragezeichen geschrieben, und der Computer wartet auf eine vom Benutzer eingegebene Zahl, die dann als Wert A zugewiesen wird. Im zweiten Beispiel wird die (wahlweise beigefügte) Zeichenkette so wie sie ist geschrieben. Sodann wird gewartet, bis der Benutzer eine Zahl eingibt, die B als Wert zugewiesen wird. Danach muß eine Zeichenkette eingegeben werden, die C$ als Wert zugewiesen wird. Die Eingaben werden durch ein Komma getrennt. Mehrfache Eingaben zu INPUT können auch durch Drücken der RETURN - Taste getrennt werden.

INT (NUM)

Das Argument NUM wird abgerundet auf die nächste ganze Zahl, die kleiner oder gleich NUM ist. Wenn NUM z.B. 2.389 ist, wird INT 2 liefern; wenn NUM –45.123345 ist, so wird der Wert von INT(NUM) –46 sein.

INVERSE

Der Videomodus wird so gesetzt, daß die **Ausgabe** des Computers in schwarzen Zeichen auf weißem Hintergrund erscheint. Mit NORMAL wird das wieder rückgängig gemacht.

LEFT$ ("APPLESOFT",5)

Die genannte Anzahl von Buchstaben, beginnend mit dem ersten in der Zeichenkette, wird als Wert geliefert. Im Beispiel kommt dabei APPLE heraus.

Linkspfeil-Taste ←

Siehe Pfeil-Tasten auf Seite 94.

LEN ("AN APPLE A DAY")
LEN (B$)

Liefert die Anzahl der Zeichen in einer Zeichenkette. Der Wert liegt zwischen Ø und 255. Im ersten Beispiel wird 14 geliefert.

A = 23.567
A$ = "DELICIOUS"

Der Variablen, die links von „=" steht, wird der Wert des Ausdrucks rechts als neuer Wert zugewiesen. Der alte Variablenwert wird überschrieben.

LIST
LIST 2ØØ, 3ØØØ
LIST 2ØØ – 3ØØØ

Im ersten Beispiel wird das gesamte Programm auf dem Bildschirm aufgelistet. Im zweiten und dritten werden nur die Zeilen von der Nummer 2ØØ bis 3ØØØ gelistet. Um von Anfang an bis zur Zeile 2ØØ zu listen, schreibt man LIST –2ØØ; um von Zeile 2ØØ bis zum Ende des Programms zu listen, schreibt man LIST 2ØØ-. Durch CTRL-C wird eine Auflistung abgebrochen. Dabei kann sie mit CONT nicht mehr erneut aufgenommen werden. Um eine Auflistung zeitweilig anzuhalten, drückt man CTRL-S. Ein erneutes CTRL-S führt die Auflistung fort.

LOAD

Ein Applesoft-Programm wird vom Kassettenrekorder eingelesen. Das Bereitschaftszeichen verschwindet. Der Benutzer muß zuerst das Band zurückspulen und die Spieltaste am Tonbandgerät drücken. Ein Piepen zeigt an, daß eine verwertbare Information auf dem Band gefunden worden ist. Wenn das Einlesen in den internen Speicher erfolgreich abgeschlossen ist, ertönt ein zweites Piepen, und das Bereitschaftszeichen (]) von Applesoft wird geschrieben. Ein LOAD kann nur durch RESET unterbrochen werden.

LOAD DOW JONES

Auf der Diskette auf dem angegebenen Laufwerk (oder dem Standardlaufwerk) wird ein Programm mit dem Namen DOW JONES gesucht. Wird das Programm gefunden, dann wird es in den Speicher des Apple eingelesen. Ein möglicherweise schon vorhandenes Programm im Speicher wird zuerst gelöscht. Danach wird das neue Programm eingelesen. In dieser Form, d.h. bei angegebenem Programmnamen, handelt es sich um einen DOS Befehl.

MID$ ("AN APPLE A DAY",4)
MID$ (DAY$, 4, 9)

Der angegebene Teil der Zeichenkette wird geliefert. Im ersten Beispiel werden die Zeichen vom vierten bis zum letzten geliefert, also APPLE A DAY. Im zweiten Beispiel werden die neun Zeichen, vom vierten bis zum zwölften, geliefert.

NEW

Das im Speicher befindliche Programm wird gelöscht und alle Variablen verschwinden.

NEXT

Siehe FOR auf Seite 96.

NORMAL

Der Videomodus wird auf Standard gesetzt. Weiße Zeichen werden auf schwarzem Hintergrund geschrieben, sowohl für Eingabe, als auch für Ausgabe.

NOTRACE

TRACE wird ausgeschaltet. Siehe TRACE auf Seite 101.

PDL (1)

Der gegenwärtige Wert (Ø bis 255) der spezifizierten Spielkontrolle wird geliefert: Ø bis 2 spezifizieren gültige Spielkontrollen.

PLOT 1Ø, 2Ø

Im Grafikmodus mit niedriger Auflösung wird ein Punkt an der angegebenen Stelle gezeichnet. Im Beispiel wird der Punkt an der Stelle X = 1Ø, y = 2Ø gezeichnet. Die Farbe des Punkts richtet sich nach dem gegenwärtigen Wert von COLOR, der Null (Schwarz) ist, falls COLOR seit dem GR Befehl nicht mehr zugewiesen wurde.

PRINT
PRINT A$; "X" = ";X

Im ersten Beispiel wird auf dem Schirm eine neue Zeile begonnen, gefolgt von der Ausführung eines RETURN. Einzelne Dinge in der Liste, die dem PRINT folgt, werden durch Komma getrennt, falls sie in verschiedene TAB Felder geschrieben werden sollen. Die Trennung durch Semikolon bewirkt, daß sie ohne Zwischenraum nacheinander geschrieben werden. Wenn A$ den Wert „CORE" und X den Wert 3 hat, wird im zweiten Beispiel

COREX = 3

geschrieben.

REM DIES IST EIN KOMMENTAR

Hiermit kann kommentierender Text dem Programm beigefügt werden.

REPT

Wenn Sie gleichzeitig die **REPT** Taste und irgendeine andere Taste drücken, bewirkt dies dasselbe, wie wenn Sie die andere Taste wiederholt gedrückt hätten. REPT kürzt ,,REPeaT'' ab, also ,,Wiederholung''.

RETURN

Verzweigt zur Zeile, die dem zuletzt ausgeführten GOSUB folgt (zu dem noch kein RETURN ausgeführt wurde).

RIGHT$ (''SCRAPPLE'', 5)
RIGHT$ (S$, 2)

Die angegebene Zahl von Zeichen am rechten Ende der Zeichenkette wird als Wert geliefert. Im ersten Beispiel kommt dabei ,,APPLE'' heraus.

Rechtspfeil Taste →

Siehe Pfeil-Tasten auf Seite 94.

RND (5)

Eine Zufallszahl, größer oder gleich Ø und kleiner als 1, wird geliefert. RND (Ø) wiederholt die zuletzt erzeugte Zufallszahl. Jedes **negative** Argument erzeugt eine gewisse Zufallszahl, die bei jedem Aufruf mit demselben Argument gleich ist. Darauffolgende RND Aufrufe mit positiven Argumenten ergeben eine wiederholbare Folge von Zufallszahlen. Jedesmal wenn RND mit einem positiven Argument aufgerufen wird, wird eine **neue** Zufallszahl erzeugt, die Teil einer unwiederholbaren Zufallsfolge sein wird, es sei denn die Folge wurde mit einem RND Aufruf mit negativem Argument begonnen.

RUN 5ØØ

Alle Variablen, Pointer und Stacks werden auf Null gesetzt. Eine Programmausführung wird eingeleitet, beginnend mit der angegebenen Zeilennummer, in unserem Beispiel also mit Zeile 5ØØ. Wenn keine Zeilennummer angegeben wird, so fängt die Ausführung mit derjenigen Zeile an, die die niedrigste Nummer hat.

RUN ANNUITY

Die Programmdatei ANNUITY wird von der Diskette auf dem angegebenen oder dem Standardlaufwerk geladen. Danach wird das Programm ausgeführt. In dieser Form, d.h. mit nachfolgendem Dateinamen, handelt es sich um einen DOS-Befehl, nicht um einen Applesoft Befehl.

SAVE

Das Programm, das sich gerade im Speicher befindet, wird auf dem Kassettenrekorder gespeichert. Kein Bereitschaftszeichen oder anderes Signal wird gegeben: Der Benutzer muß zuerst die Aufnahme- und Abspieltaste drücken, bevor SAVE ausgeführt wird. SAVE wird diesen Sachverhalt nicht nachprüfen. Ein Piepsen der Maschine zeigt Anfang und Ende des Aufnahmevorgangs an.

SAVE ADDRESSES

Die Datei, die sich gerade im Speicher befindet, wird auf der Diskette abgespeichert. Wenn keine Datei mit gleichem Namen auf der Platte, die sich auf dem spezifizierten oder Standardlaufwerk befindet, gefunden wird, so wird eine neue Datei mit diesem Namen geschaffen, und das Programm im Speicher darin eingeschrieben. Wenn schon eine Datei dieses Namens vorhanden ist, und wenn sie den gleichen Typ hat, so wird die alte Datei überschrieben und ist nun verloren. Keinerlei Warnungen werden gegeben. SAVE ist in dieser Form ein DOS Befehl.

STR$ (12.45)

Die Zeichenkette, die den Wert des Arguments darstellt, wird geliefert. In diesem Beispiel wird „12.45" geliefert.

TAB (23)

Der Befehl darf nur im PRINT Befehl verwandt werden. Das Argument muß einen Wert zwischen Ø und 255 haben und in Klammern eingeschlossen werden. Für Argumentwerte von 1 bis 255 wird TAB den Cursor von links gezählt in die entsprechende Schreibposition bringen, wobei der Wert größer als die gegenwärtige Cursor Position sein muß. Wenn der Argumentwert kleiner als die augenblickliche Cursor-Position ist, so geschieht nichts. TAB (Ø) versetzt den Cursor in Position 256.

TEXT

Der Bildschirm wird in den Standardmodus versetzt, also nicht in Grafikmodus. Dadurch werden 24 Zeilen zu je 4Ø Buchstabenpositionen darstellbar. Anders gesagt, das Textfenster wird auf den gesamten Bildschirmbereich erweitert.

TRACE

Die Zeilennummer jeder Zeile, die gerade ausgeführt wird, wird auf dem Bildschirm geschrieben, TRACE wird **nicht** durch RUN, CLEAR, NEW, DEL oder RESET abgeschaltet. NOTRACE schaltet TRACE ab.

VAL ("-3.7E4A5PLE")

Die Zeichenkette wird als Zahl aufgefaßt, und zwar bis zum ersten Buchstaben von links. Dieses Präfix wird dann in eine ganze Zahl oder relle Zahl konvertiert und als numerischer Wert geliefert. Wenn keinerlei Zahlen am Anfang der Kette stehen, wird Ø geliefert. In diesem Beispiel wird als Wert -37ØØØ geliefert, da -3.7E4 diese Zahl abkürzt.

VLIN 1Ø, 2Ø AT 3Ø

Im Grafikmodus mit niedriger Auflösung wird eine vertikale Linie gezeichnet. Die Farbe der Linie richtet sich nach der letzten Wertzuweisung von COLOR. Die Spalte, in der die Linie erscheint, wird als drittes Argument (AT folgend) angegeben, Anfangs- und Endposition durch das erste und zweite Argument. In unserem Beispiel wird die Linie von X = 3Ø, y = 1Ø nach X = 3Ø, y = 2Ø gezeichnet.

VTAB (15)

Der Cursor wird auf die angegebene Zeile des Bildschirm gebracht. Die oberste Zeile hat die Nummer 1, die unterste 24. VTAB bewegt den Cursor nach oben oder nach unten, aber niemals nach rechts oder nach links.

ANHANG B :

Reservierte Worte im Applesoft

Die folgende Liste enthält **alle** im Appelsoft BASIC reservierten Worte. Obwohl wir viele dieser Worte nicht in diesem Buch besprochen haben, wird diese Liste Ihnen nützlich sein, wenn Sie Variablen benennen wollen. Im Applesoft BASIC Programmierhandbuch finden Sie mehr über die Benutzung dieser Befehlsworte.

&

ABS
AND
ASC
AT
ATN

CALL
CHR$
CLEAR
COLOR =
CONT
COS

DATA
DEF
DEL
DIM
DRAW

END
EXP

FLASH
FN
FOR
FRE

GET
GOSUB
GOTO
GR

HCOLOR =
HGR
HGR 2
HIMEM:
HLIN
HOME
HPLOT
HTAB

IF
IN#
INPUT
INT
INVERSE

LEFT$
LEN
LET
LIST
LOAD
LOG
LOMEM :

MID$

NEW
NEXT
NORMAL
NOT
NOTRACE

ON
ONERR
OR

PDL
PEEK
PLOT
POKE
POP
POS
PRINT
PR#

READ
RECALL
REM
RESTORE
RESUME
RETURN
RIGHT$
RND
ROT =
RUN

SAVE
SCALE =
SCRN (
SGN
SHLOAD
SIN
SPC (
SPEED =
SQR
STEP
STOP
STORE
STR$

TAB (
TAN
TEXT
THEN
TO
TRACE

USR

VAL
VLIN
VTAB

WAIT

XPLOT
XDRAW

Appelsoft benutzt einen besonderen Kode für diese Namen, so daß jeder genau ein Byte Speicherplatz beansprucht. Sonstige Buchstaben im Programm nehmen je ein Byte Speicherplatz ein.

Das „et" Zeichen (&) ist für den internen Gebrauch im Computer bestimmt und ist **kein** erlaubter Applesoft-Befehl. Als Befehl ausgeführt, bewirkt dieses Symbol eine unbedingte Verzweigung nach der Adresse $3F5.

XPLOT ist ein Wort, das bislang keinem Applesoft-Befehl entspricht, das aber für spätere Verwendung reserviert worden ist.

Einige dieser reservierten Worte sind für Applesoft nur in bestimmten Zusammenhängen erkennbar.

COLOR, HCOLOR, SCALE, SPEED, und ROT

sind nur dann als reservierte Worte kenntlich, wenn der nächstfolgende Buchstabe ein = ist (Leerzeichen werden ignoriert). Im Falle von HCOLOR und COLOR ist das eine unwesentliche Einschränkung, denn wegen des reservierten Wortes OR kann man sie Ohnehin nicht als variablen benutzen.

SCRN, SPC und TAB

Sind nur dann als reservierte Worte kenntlich, wenn der nächstfolgende Buchstabe eine Klammer-auf ist, (.

HIMEM : Der Doppelpunkt (:) muß folgen, damit es als reserviertes Wort erkannt wird.

LOMEM : Wie bei HIMEM: muß der Doppelpunkt folgen, sonst wird es nicht als reserviertes Wort angesehen.

ATN wird nur als reserviertes Wort betrachtet, wenn kein Zwischenraum zwischen dem T und dem N ist. Wenn ein Zwischenraum vorhanden ist, so wird AT als das reservierte Wort AT betrachtet, statt ATN zu lesen.

TO ist immer reserviertes Wort, **außer** vor dem TO steht ein A **und** zwischen T und O ist ein Zwischenraum. In dem Falle wird AT anstatt TO gelesen.

Manchmal kann man Klammern dazu benutzen, daß nicht ungewollt reservierte Worte gelesen werden. Zum Beispiel

```
                      100 FOR A = LOFT OR CAT TO 15
wird aufgelistet als  100 For  A = LOF TO RC AT TO 15
                aber  100 FOR A = (LOFT) OR (CAT) TO 15
wird aufgelistet als  100 FOR A = (LOFT) OR CC AT) TO 15
```

ANHANG C:
Aufbereitungsmittel

Links- und Rechtspfeil-Taste

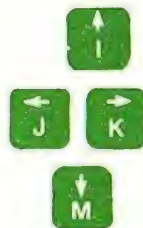
Die Linkspfeil-Taste, auch BACKSPACE-Taste genannt, bewegt dan Cursor um eine Stelle nach links. Der dabei übersprungene Buchstabe wird gelöscht. Falls Sie noch nicht zum Abschluß der zuletzt getippten Zeile die RETURN -Taste gedrückt haben, hat die Taste nur auf die Buchstaben dieser Zeile eine Wirkung.

Durch Drücken der Rechtspfeil-Taste die auch RETYPE-Taste genannt wird, bewegt sich der Cursor vorwärts (d.h. nach rechts). Jeder übersprungene Buchstabe wird neu eingegeben. Wenn Sie mit dieser Taste über eine Zeile hinweggehen und dann die RETURN - drücken, so verhält sich der Apple so als hätten Sie die Zeile erneut getippt.

Mittels der REPT -Taste kann man die Pfeiltasten schnell wiederholen, indem man sie gleichzeitig mit den Pfeiltasten drückt.

Reine Cursor-Bewegungen

Die Tasten ESC, I, J, K und M werden benutzt, um den Cursor über den Bildschirm zu bewegen, ohne daß dabei etwas mit den übersprungenen Buchstaben geschied. Stellen Sie sich vor, auf den Tasten wären Pfeile gemalt :



Die ESC-Taste versetzt Sie in Aufbereitungsmodus. Sobald Sie in diesem Modus sind, wird sich der Cursor durch eine der vier Tasten in die gewünschte Richtung bewegen. Sie dürfen den Cursor an jede Stelle auf dem Bildschirm bringen.

Um den Cursor schneller in eine bestimmte Richtung zu bewegen, können Sie gleichzeitig die REPT-Taste drücken. Dadurch bewegt sich der Cursor mit einiger Geschwindigkeit in die gewünschte Richtung. Wenn der Cursor die Oberkante des Bildschirms erreicht, hält er an. Erreicht er die Unterkante, so hält der Cursor selbst zwar an, jedoch werden daraufhin die Zeilen nach oben gerollt. Wenn der Cursor die rechte Bildschirmkante erreicht, dann verschwindet er zunächst, erscheint aber dann auf der linken Kante, und zwar auf der nächsten Zeile. An der linken Kante verschwindet er und taucht dann eine Zeile höher rechts wieder auf. Um in den normale textmodus zurückzukehren, drückt man die Leertaste.

Löschen von Programmzeilen

Als einfachste Art, eine Zeile aus einem Programm zu löschen, schreibt man ihre Nummer und drückt die RETURN-Taste. Wenn mehr als eine Zeile gelöscht werden soll, so könnten Sie den DEL-Befehl benutzen. Um etwa Zeilen 100 bis 200 zu löschen, schreiben Sie

DEL 100, 200

Dadurch werden alle Zeilen, deren Nummern zwischen 100 und 200 einschließlich liegen, gelöscht.

Die Tastenkombination



wird die gerade geschriebene Zeile löschen. Das ist dann nützlich, wenn Sie vor dem Druck auf die RETURN-Taste plötzlich bemerken, daß Sie sich verschrieben haben.

Löschen des Bildschirms

Die folgenden Befehle verändern nur, was Sie auf dem Bildschirm sehen. Was im Speicher des Computers gerade steht, bleibt dabei unverändert. Ein einzelner Druck auf die ESC -Taste versetzt Sie in Aufbereitungsmodus.

Um den Bildschirm zu löschen, drücken Sie zuerst ESC und dann den „Klammeraffen" @ :



Der Cursor wird auf die linke obere Ecke des Bildschirms gesetzt, ohne daß ein Bereitschaftszeichen erscheint. Das Bereitschaftszeichen erhalten Sie durch ein RETURN.

Wenn Sie schon im Aufbereitungsmodus sind, reicht es natürlich, bloß den Klammeraffen zu tippen. Der Apple wird außerdem in den normalen Textmodus rückversetzt.

Der HOME-Befehl löscht übrigens auch den Bildschirm. Schreiben Sie einfach

HOME

dann läuft der Cursor „nach Hause" in die Ecke links oben.

Es ist möglich, den Bildschirms teilweise zu löschen. Um den Bildschirm von einem bestimmten Punkt an bis zum Bildschirmende zu löschen, gehen Sie zunächst in Aufbereitungsmodus per ESC-Taste. Danach bewegen Sie den Cursor an den gewünschten Anfangspunkt unter Benutzung der Tasten I, J, K und M. Drücken Sie nun die F-Taste, dann verschwindet alles vom Anfangspunkt an bis zur Unterkante des Bildschirms. Um nur bis zum Zeilenende zu löschen, Drücken Sie die E-Taste. In beiden Fällen werden Sie außerdem in den normalen Textmodus zurückversetzt.

Zusammenfassung der Aufbereitungsmittel

Aufbereitungsmodus herstellen	Drücke ESC
Aufbereitungsmodus beenden	Drücke dit Leertaste
Den Cursor bewegen	Drücke I J K oder M
Einen Buchstaben löschen	Drücke ←
Einen Buchstaben erneut schreiben	Drücke →
Von der Cursor Position bis zum Zeilenende löschen	Drücke ESC und dann E
Von der Cursor Position bis zur Bildschirm Unterkante löschen	Drücke ESC und dann F
Den gesamten Bildschirm löschen	Drücke ESC und dann sowohl CTRL , als auch @ P
Auflistung zeitweilig anhalten	Drücke CTRL und S
Mit der Auflistung weitermachen	Drücke CTRL und S

ANHANG D :
Unterschiede zwischen Firmaware-, kassetten- und Disketten-Version des Applesoft

Einleitung

Sie brauchen diesen Anhang überhaupt nicht zu lesen, es sei denn, Sie benutzen eins fer folgenden Dinge :

1. Die Applesoft II Firmwarekarte zum Einstecken.
2. Applesoft, das vom Kassettenrekorder geladen wird.
3. Applesoft, das von der Diskette geladen wird.

Wenn Sie zum ersten Mal mit Computern arbeiten, mag Ihnen manches in diesem Anhang sehr technisch erscheinen. Es macht nichts, wenn Sie nicht alles auf Anhieb verstehen. Lesen Sie sich durch, was für Sie zuzutreffen scheint, und holen Sie sich heraus, was Sie an Information brauchen. Zu einem späteren Zeitpunkt, wenn Sie mehr von Computern verstehen, können Sie diesen Anhang dann nochmal in aller Ruhe durchlesen, um die feineren Nuancen zu verstehen.

Manche Bezeichnugen in diesem Anhang mögen Ihnen fremd erscheinen. Viele werden den **Speicher** des Apple beschreiben, der auf verschiedenste Art benutzt wird :

1. Um die Kassetten- oder die Diskettenversion des Applesoft zu speichern.
2. Um die Befehle, die Ihr Programm darstellen, zu speichern.
3. Um die Variablen und Zeichenketten zu speichern, die Ihr Programm manipuliert.
4. Verschiedene Information zu speichern, die der Apple selbst braucht, etwa wo im Speicher was zu finden ist, Daten über Ihr Programm, über das Betriebssystem, usw.
5. Um den Text und die Grafiken mit niedriger Auflösung zu erzeugen, damit sie auf dem Fernsehbildschirm dargestellt werden können.
6. Um Grafiken mit hoher Auflösung zu erzeugen, die auf dem Bildschirm erscheinen sollen.

Alle diese Dinge und Aktivitäten beanspruchen Speicherplätze, die im allgemeinen an unterschiedliche Stellen im Speicher gelegt werden. Die Daten werden in „Kästchen" im Speicher abgelegt. So ein Kästchen nennt man **Wort**. Ein Block von 1024 wird manchmal **1 K** Speicher genannt. Jedes hat eine **Adresse**, einfach eine Zahl, so daß eine bestimmte Stelle im Speicher wiedergefunden werden kann. In unserem Falle wird das Speicherwort **Byte** genannt. Sie werden sehr selten Speicherinhalte direkt zu sehen bekommen. Statt dessen wird Ihnen dargestellt, was ein oder mehrere Speicherworte **kodieren,** denn direkt kann man im Speicher nur ganze Zahlen aufbewahren.

Denjenigen Teil des Speichers, der für bestimmte Aktivitäten benutzt wird, werden wir zunächst durch Adressen beschreiben. In den meisten Fällen wird es sich da um eine Anfangs- und eine Endadresse handeln. Wenn z.B. Ihr Programm einen bestimmten Speicherabschnitt benutzt, dann darf dieser Abschnitt nicht gleichzeitig für eine andere Aktivität (etwa Grafikerstellung) benutzt werden, sonst wird Ihr Programm teilweise oder ganz verloren gehen.

In Applesoft BASIC drücken wir Zahlen und Adressen in **Dezimalform** aus. Die Maschine benutzt aber intern eine andere Zahlendarstellung, die wir **Hexadezimal** nennen. Um fortgeschrittenen Programmierern entgegenzukommen, drücken wir manchmal Speicheradressen sowohl dezimal als auch hexadezimal aus. Dabei kennzeichnen wir die hexadezimalen Zahlen dadurch, daß wir ein Dollarzeichen ($) davorsetzen. Wenn Sie damit nichts anfangen können, dürfen Sie solche Zahlen getrost ignorieren.

Allgemeine Bemerkungen

Die Firma Apple bietet zwei Versionen der Programmiersprache BASIC an. Integer BASIC, die eine Version, die auch im Apple II BASIC Programmierhandbuch beschrieben ist, ist eine sehr schnelle Programmiersprache, die sich für viele Anwendungen gut eignet, besonders für Lehrzwecke, Computerspiele und Grafiken. Die andere Version nennt sich **Applesoft** und eignet sich besser für naturwissenschaftliche und geschäftliche Anwendungen.

Das Applesoft BASIC ist wiederum in zwei Formen zu haben: **Firmware Applesoft** und **Disketten oder Kassettenrekorderversion**. Das Firmware-Appelsoft kommt in Form einer sogenannten ROM (Read Only Memory), einem permanenten Speicher, aus dem man lesen, in den man aber nicht schreiben kann. Diese ROM Chips können in die DØ bis FØ markierten Stecker eingestöpselt werden, oder sie kommen auf der sogenannten Firmware-Karte (Nr. A2BØØØ9X), die auch eingestöpselt wird. Dieses Firmware-Applesoft ist sofort da, wenn Sie Ihren Apple einschalten, oder wenn Sie den Befehl FP geben. Das erspart Ihnen die Zeit, die sonst dazu gebraucht wird, um das Applesoft von Band oder Diskette in den Speicher zu laden. Außer diesem Luxus werden etwa 1ØK Speicherplatz frei, die sonst für Applesoft gebraucht würden.

Im größten Teil dieses Buches haben wir angenommen, daß Sie die Firmware-Version installiert haben, und zwar eingestöpselt in die mit DØ bis FØ markierten Stecker. Teil 1 dieses Anhangs erläutert, wie die Applesoft II Firmwarekarte installiert und gebraucht wird.

Wenn Sie aber die Disketten- oder Bandversion des Applesoft haben, dann markieren die Symbole



diejenigen Stellen im Buch, an denen Ihre Version von der Firmware Version abweicht. Im Teil 2 dieses Anhangs besprechen wir die Diskettenversion des Applesoft, in Teil 3 die Bandversion. In Teil 4 fassen wir die besonderen Hinweise und Informationen über die Unterschiede der einzelnen Versionen zusammen, die wir verstreut im Hauptteil des Buches schon angaben. Und schließlich geben wir für den forgeschrittenen Programmierer in Teil 5 an, wie der Speicherplatz im Appelsoft aufgeteilt ist.

Ein wichtiger Hinweis

Das Bereitschaftszeichen hat nicht nur die Funktion, Ihnen zu signaliseren, daß die Maschine für Ihre Eingabe bereit ist, sondern soll auch kenntlich machen, in welcher Programmiersprache das sein muß. Sie werden wahrscheinlich die folgenden Zeichen im Lauf der Zeit zu Gesicht bekommen:

* Für Monitor-Programme. Bekommt man auf CALL - 151 hin.

> Für Aplle-Integer-BASIC (Ganzzahl-Basic).

] Für Applesoft (floating point) BASIC (Fließkomma-Basic).

Auf diese Weise sehen Sie auf einen Blick, welche Sprache die Maschine gerade versteht.

TEIL 1 :
Die Applesoft Firmaware-Karte

Installierung

Die Karte wird einfach in den Stecker innerhalb des Apple eingestöpselt. Beachten Sie aber dabei genau das Folgende :

1) **Schalten Sie den Apple ganz ab**. Das ist überaus wichtig. Wenn Sie es versäumen, kann das Gerät Schaden nehmen.

2) Nehmen Sie den Verschlußdeckel des Apple ab. Das tun Sie, indem Sie hinten am Deckel nach oben ziehen, also an der von den Tasten am weitesten entfernten Kante. Dabei öffnen sich die Klammern, die den Deckel festhalten. Ziehen Sie den Deckel nicht weiter nach oben, sondern schieben Sie ihn nach hinten, bis er freigekommen ist.

3) Im Inneren des Apples entlang des hinteren Teils des großen Schaltbretts befindet sich eine Reihe von acht langen und feinen Steckern. Der Stecker ganz links (von der Tastatur aus gesehen) ist stecker #Ø, derjenige ganz rechts ist Stecker #7. Sie halten die Firmware-Karte nun so, daß der kleine Schalter auf ihr nach hinten zeigt. Sie Schieben danach das Steckende der Karte in den Stecker ganz links, also den Stecker #Ø. Das wird nicht ganz ohne Widerstand gehen, jedoch wird anschließend die Karte sicher im Stecker sitzen. Die Firmwarekarte **muß** in den Stecker #Ø geschoben werden!

4) Der Schalter auf der Firmaware-Karte sollte nun teilweise aus dem Schiltz im hinteren Gehaüse herausragen.

5) Setzen Sie den Deckel wieder auf den Apple auf. Zuerst wird er nach vorne hereingeschoben und dann an der Hinterkante ins Gehäuse gedrückt bis die Klammern einschnappen.

6) Nun können Sie Ihr Gerät wieder an das Netz anschließen und einschalten.

Gebrauch der Applesoft II Firmware-Karte

Wenn der Schalter auf der Firmaware-Karte nach **unten** zeigt, wird Ihr Apple beim Einschalten auf Integer BASIC ansprechen. Sie sehen das am Bereitschaftszeichen >, welches ja das Integer BASIC identifiziert.

Wenn der Schalter nach **oben** zeigt, wird Applesoft BASIC statt Integer BASIC laufen. Das Bereitschaftszeichen] zeigt Ihnen diesen Sachverhalt an.

Wenn Sie das DOS-Betriebssystem benutzen, wird der Computer unabhängig von der Schalterstellung entweder Applesoft BASIC oder Integer BASIC automatisch wählen, je nach Bedarf.

TEIL 2:
Die Diskettenversion des Applesoft

Mit jedem Disk II wird Applesoft II BASIC auf der Integer Basic System-Masterdiskette geliefert, und zwar als Programm mit dem Namen APPLESOFT.

Der Apple selbst braucht 2K Speicherplatz, um interne Daten zu speichern. Das DOS Betriebssystem braucht 10.5K und das Applesoft BASIC von der Diskette weitere 10K Bytes. Daher brauchen sie **mindestens** 24K Speicherplatz insgesamt, bevor Sie die Diskettenversion des Applesoft gebrauchen können.

Um die Diskettenversion zu gebrauchen, müssen Sie zuerst einmal die Diskette zum Laufen bringen („boot"), und Sie müssen natürlich die Diskette einlegen, die das APPLESOFT überhaupt enthält, also die System-Masterdiskette. Sie Können Applesoft **nicht** durch **RUN APPLESOFT starten**. Dieser Befehl wird das Programm nicht richtig initialisieren, denn obwohl alles zunächst richtig aussehen wird, werden Sie sofort in Schwierigkeiten geraten, sobald Sie einen DOS Befehl geben oder die RESET-Taste drücken. Sie müssen vielmehr den Befehl

FP

(Abkürzung für „Floating Point Basic") zum Starten des Applesoft benutzen. Sobald Sie diesen Befehl

FP

geben, wird Ihr Computer versuchen, das APPLESOFT Programm von der Diskette zu laden, die sich auf dem Standardlaufwerk befindet (also das zuletzt benutzte). Falls dieses Programm nicht zu finden ist, wird die Nachricht:

LANGUAGE NOT AVAILABLE

erscheinen. Wenn das geschieht, haben Sie zwei Möglichkeiten: Sie können die richtige Diskette in das Standard-Gerät legen und erneut

FP

befehlen, oder, wenn Sie wissen, daß die richtige Diskette auf einem anderen Laufwerk ist, können Sie den FP-Befehl mit zusätzlichem Laufwerksparameter geben. Wenn z.B. die Diskette in dem Laufwerk 2 ist, und wenn das Laufwerk über die Controller Karte im Stecker #6 angeschlossen worden ist, schreiben Sie

FP, S6, D2

Weitere Hinweise ersehen Sie aus dem DOS Handbuch.

Wenn Sie ein Programm mit LOAD von der Diskette einlesen und mit RUN laufen lassen, wird das DOS automatisch in die korrekte Sprache umschalten. Wenn es sich dabei um Applesoft handelt, wird DOS versuchen das APPLESOFT Programm zu finden. DOS wird auf derjenigen Diskette suchen, die sich in dem Laufwerk, das im LOAD oder RUN Befehl angegeben wurde, befindet. Falls kein Laufwerk angegeben wurde, wird auf dem Standardlaufwerk gesucht. Natürlich können Sie auch selbst mit dem FP Befehl auf Applesoft umschalten.

TEIL 3 :
Die Kassettenrekorder-Version des Applesoft

Mit jedem Apple II wird Applesoft BASIC in der Kassettentonbandversion mitgeliefert. Wenn Sie außerdem die Firmware-Karte oder ein Floppy-Laufwerk haben, werden Sie die Bandversion nicht gebrauchen. Wenn Applesoft BASIC vom Tonband geladen wird, benötigt es etwa 10K Bytes Speicherplatz. Der Apple selbst benötigt weitere 2K, um den Bildschirm usw. zu verwalten. Daher brauchen Sie mindestens 16K Bytes Speicherplatz, bevor Sie die Tonbandversion gebrauchen können.

Wie Applesoft von der Kassette geladen wird

Sie laden Applesoft BASIC vom Kassettenrekorder wie folgt :

1) Starten Sie Integer BASIC, indem Sie den Computer anschalten. Sie wissen, daß das geschehen ist, sobald das Bereitschaftszeichen > auf dem Bildschirm, gefolgt vom blinkenden Cursor, erscheint.

2) Legen Sie die Applesoft Kassette in das Tonbandgerät, und spulen Sie sie zum Bandanfang zurück. Die Kassette trägt die Nummer A2T0004.

3) Schreiben Sie LOAD.

4) Drücken Sie die Abspieltaste auf Ihrem Tonbandgerät.

5) Drücken Sie sofort die **RETURN** Taste. Sobald das geschehen ist, wird der Cursor vom Bildschirm verschwinden. Nach etwa 5 bis 20 Sekunden wird der Apple piepen, um anzuzeigen, daß das Programm eingelesen wird. Nach etwa ein bis eineinhalb Minuten wird die Maschine erneut piepen, und der Cursor und das Bereitschaftszeichen > werden erneut erscheinen.

6) Halten Sie das Tonbandgerät an, und spulen Sie das Band zurück. APPLESOFT befindet sich jetzt im Speicher des Computers.

7) Schreiben Sie RUN, und drücken Sie die **RETURN**-Taste. Nun wird auf dem Bildschirm die Copyrightzeile des Applesoft erscheinen und dann das Bereitschaftszeichen des Applesoft].



Wenn das Monitor-Programm läuft, d.h. das Bereitschaftszeichen * zu sehen ist, dann wird die Tastenkombination **CTRL-B** Sie auf Integer BASIC umschalten. Dadurch wird Applesoft gelöscht.

TEIL 4:
Unterschiede zwischen Firmaware-, Kassetten- und Disketten-Version des Applesoft

Das Appelsoft in der Kassetten-und der Diskettenversion funktioniert etwas anders als die Firmwareversion auf der Firmware-Karte (Nr. A2B0009X) oder die ROM-Version, die in die Stecker D0 bis F0 eingesteckt wird. Wir haben hier im Wesentlichen die Firmware-Version beschrieben. Wir werden nun erläutern, worin die Unterschiede zwischen den verschiedenen Versionen bestehen.

Da die Firmwareversion keinen Speicherplatz benötigt, kann sie bei praktisch jeder Speicherkonfiguration verwendet werden.

Diskette-Applesoft benötigt etwa 10K Bytes Speicher, der Apple braucht weitere 2K, um den Bildschirminhalt zu speichern, und das DOS Betriebssystem benötigt zusätzlich 10.5K Bytes. Man kann daher diese Version in Speicherkonfigurationen von unter 24K Bytes nicht verwenden. Bei geladenem Applesoft (Diskettenversion) ist die erste verfügbare Speicheradresse 12291. Beachten Sie dazu auch das Speicherzuordnungsdiagramm auf Seite 114.

Die Tonbandversion benötigt ebenso etwa 10K Bytes Speicherplatz, und der Apple weitere 2K Bytes. Daher kann diese Version nicht in Speicherkonfigurationen unter 16K benutzt werden. Wenn die Bandversion des Applesoft geladen ist, so ist 12291 die erste verfügbare Speicheradresse. Sehen Sie auch dazu das Speicherzuordnungsdiagramm auf Seite 114 ein.



Der HGR Befehl ist nicht in der Platten- oder Bandversion des Applesoft verfügbar. Der HGR Befehl löscht nämlich „Seite 1" des Grafikspeichers, der im Speicher auf Adresse 8192 - 16383 liegt. Da die Disketten- und die Bandversion auch teilweise diesen Speicherabschnitt beanspruchen, wird der HGR Befehl das Applesoft-Programm löschen. Damit geht auch Ihr Programm verloren. Sie dürfen HGR also nur in der Firmware-Version des Applesoft benutzen.

Der HGR2 Befehl benutzt „Seite 2" des Grafikspeichers, die dem Speichersegment von 16384 - 24575 zugeordnet ist. HGR2 kann sowohl in der Firmware- als auch in der Tonbandversion des Applesoft benutzt werden. Dazu brauchen Sie aber mindestens 24K Speicherplatz. Wenn Sie also die Bandversion haben, und Sie haben weniger als 24K Speicherplatz, so können Sie keine Grafiken mit hoher Auflösung haben.



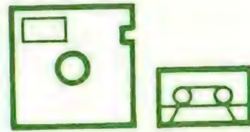
In der Diskettenversion des Applesoft und in der Firmware Version unter Benutzung des DOS Systems, könnte HGR2 Schwierigkeiten verursachen. Da es die „Seite 2" löscht, also den Adressbereich von 16384 bis 24575, würden Sie bei weniger als 36K Bytes Speicherplatz große Teile des DOS verlieren. Wenn Sie also weniger als 36K Bytes Speicher haben und die Diskettenversion des Applesoft benutzen (oder die Firmware Version und das DOS System), können Sie keine Grafiken mit hoher Auflösung haben.

Der Befehl

POKE - 16.301,0

wird im Applesoft benutzt, um Grafik mit Text mischen zu können. Wenn dieser Befehl auf HGR 2 folgt, so werden allerdings die vier Zeilen Text im Bildschirm unten von der **Seite 2** des Bildschirmspeichers gelesen, und nicht, wie gewöhnlich, von der Seite 1 des Speichers. Die Disketten- und die Bandversion des Applesoft benutzten jedoch auch die Seite 2 des Speichers, so daß Grafik mit Text vermischt in dieser Version nicht möglich ist.

In der Platten- oder Bandversion des Applesoft können Sie CALL 11246 (statt des CALL 62450 der Firmware Version) benutzen, um den HGR2 Bildschirm auf Schwarz zu löschen. Um den Bildschirm zu der Farbe HCOLOR bei HGR2 zu löschen, benutzen Sie CALL 11250 (statt CALL 62454 der Firmware Version).



Wenn Sie die obigen Befehle geben, ohne zunächst HGR2 zu befehlen, so werden diese Befehle statt dessen die Seite 1 des Grafikspeichers löschen und damit Ihr Applesoft.

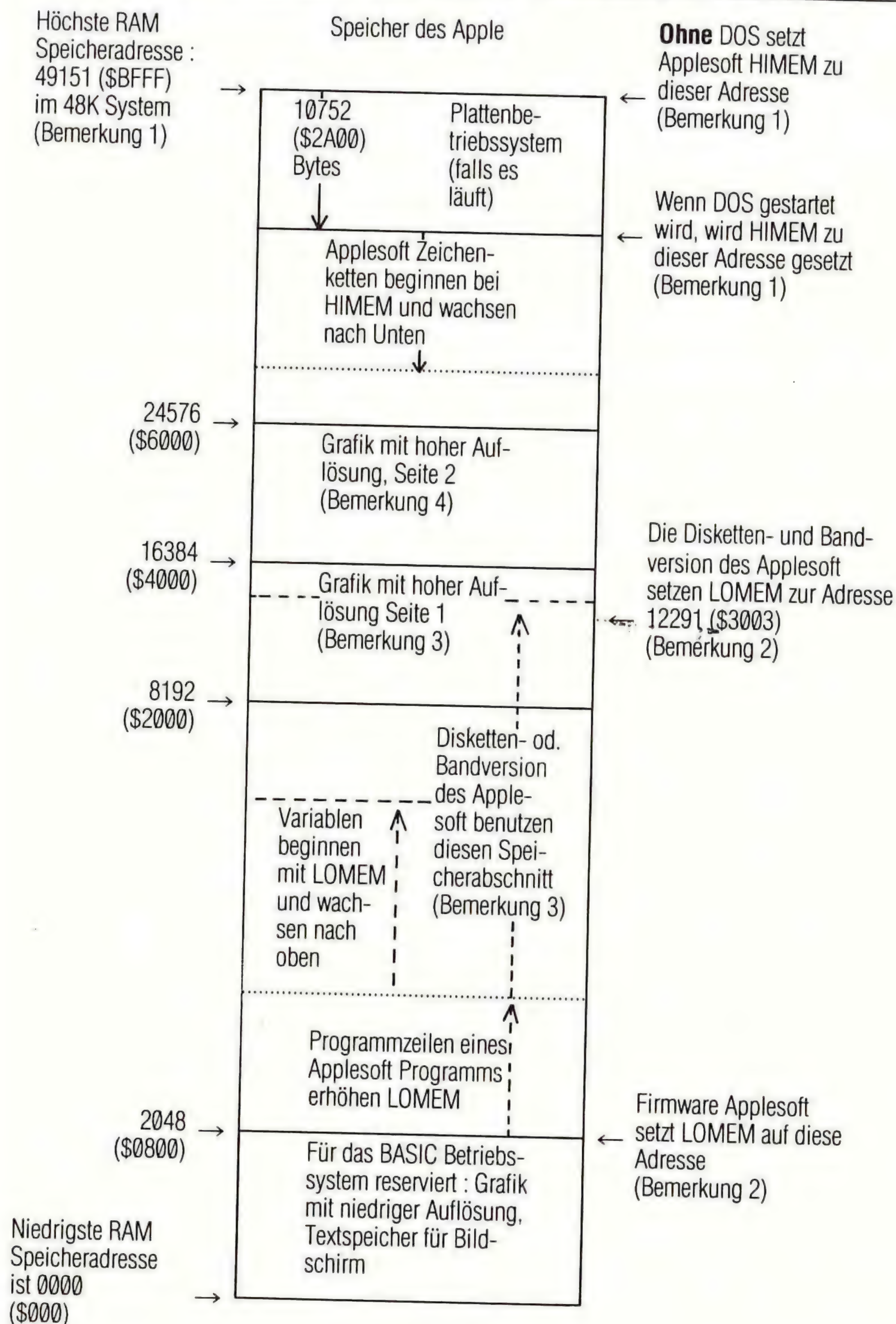
In der Frimware-Version des Applesoft können Sie HPLOT mit „TO Ketten" verwenden, also

HPLOT X1, Y1 TO X2, Y2 TO X3, Y3 TO X4, Y4

In der Diskette-Diskettenversion und in der Bandversion müssen Sie diese Kette auflösen und schreiben

HPLOT X1, Y1 TO X2, Y2
HPLOT TO X3, Y3
HPLOT TO X4,Y4

TEIL 5 : Speicherplatz-Zuweisung für DOS und für Applesoft BASIC



Bemerkung 1. HIMEM enthält die höchste Adresse im Speicher, die ein Applesoft Programm benutzen kann. Wenn auf Ihrer Maschine Applesoft läuft, können sie den Wert von HIMEM in den Adressen 115 und 116 ($73 und $74) ablesen. Diese Adresse ist hexadezimal, und die niedrigen Ziffern kommen zuerst (115). Sie erhalten den Wert dezimal, wenn Sie

```
PRINT PEEK(115) + PEEK(116) * 256
```

eingeben. Die folgende Tabelle gibt Ihnen die genauen Werte für HIMEM, wenn sie DOS laufen lassen, in Abhängigkeit der Gesamtgröße Ihres Speichers. Wenn Sie MAXFILES vergrößern, so wird HIMEM um zusätzliche 595 Bytes für jeden Dateipuffer herabgesetzt. Die Adressen, in denen Sie andere Systemparameter ablesen können, werden im Anhang I des Applesoft II BASIC Programmierhandbuchs angegeben.

HIMEM-Werte nach Starten des DOS-Betriebssystems

System	Höchste RAM Adresse		HIMEM Werte nach Start des DOS	
Größe	Dezimal	Hexadezimal	Dezimal	Hexadezimal
16K	16383	$3FFF	5632	$1600
20K	20479	$4FFF	9728	$2600
24K	24575	$5FFF	13824	$3600
32K	32767	$7FFF	22016	$5600
36K	36863	$8FFF	26112	$6600
48K	49151	$BFFF	38400	$9600

Die gleichwertige negative Adresse einer jeden positiven Adressen wird nach der Formel n – 65536 errechnet.

Bemerkung 2. LOMEM gibt die niedrigste Adresse an, die ein Applesoft-Programm verwenden kann. Dieser Wert kann an den Stellen 105 und 106 ($69-$6A, hexadezimal) gefunden werden, wenn Sie Applesoft laufen haben, und wie bei HIMEM kommen die unteren Ziffern zuerst. Sie erhalten den momentanen Wert mittels des Befehls

```
PRINT PEEK(105) + PEEK(106) * 256
```

Applesoft weist den LOMEM Wert automatisch zu, und zwar zu der Adresse, die gleich auf die letzte Programmzeile des momentan gespeicherten Programms folgt.

Bemerkung 3. Wenn Sie mit HGR Grafiken mit hoher Auflösung zeichnen, wird Seite 1 des Grafikspeichers gelöscht, also von Adresse 8192 bis 16383. Wenn Sie die Firmware-Version benutzen und DOS läuft, dann wird dabei ein Teil des DOS Programms gelöscht, es sei denn, der HIMEM-Wert, den DOS errechnete, überschreitet 16383.

Daher können Sie nicht gleichzeitig DOS benutzen und Grafiken mit hoher Auflösung zeichnen, es sei denn, Sie haben mindestens 32K Bytes zur Verfügung.

Wenn Sie die Disketten- oder Bandversion des Applesoft benutzen, wird HGR, das die Seite 1 des Grafikspeichers benutzt, immer einen Teil des Applesoft löschen. Mit der Disketten- oder Bandversion können Sie also nur Seite 2 des Grafikspeichers benutzen. Beachten Sie aber auch Bemerkung 4.

Bemerkung 4. Da HGR2, Grafik mit hoher Auflösung unter Benutzung von Seite 2 des Grafikspeichers, den Speicher von Adresse 16384 bis 24575 löscht, wird dieser Befehl Teile des DOS-Betriebssystems zerstören, es sei denn, DOS errechnet HIMEM zu einem Wert größer als 24575. Das heißt also, Sie können nicht gleichzeitig DOS und HGR2 benutzen, außer Sie haben mindestens 36K Bytes Speicherplatz zur Verfügung.

ANHANG E :
Fehlermeldungen

Alle Fehlermeldungen, die Ihnen Applesoft BASIC geben kann, sind hier mit Erklärungen gelistet. Weitere Erklärungen können Sie im Applesoft BASIC Programmierhandbuch finden.

Sobald ein Fehler auftritt, kehrt Applesoft zur unmittelbaren Ausführung zurück, was durch das Bereitschaftszeichen] und den blinkenden Cursor signalisiert wird. Obwohl Variablenwerte erhalten bleiben, kann ein Programm mit CONT nicht weitergeführt werden, und alle ausstehenden GOSUBs und FOR Schleifenzähler werden auf null gesetzt, bzw. als nicht mehr ausstehend betrachtet.

Wenn der Fehler in einem Befehl auftritt, der unmittelbar auszuführen war, so wird keine Zeilennummer geschrieben.

Die Fehlermeldung hat folgendes Format :

In unmittelbarer Ausführung	?XX ERROR
In aufgeschobener Ausführung	?XX ERROR IN YY

Wobei das ,,XX'' den vorliegenden Fehler nennt, während ,,YY'' die Programmzeile angibt, in der der Fehler bei der Programmausführung auftrat. Fehler in Programmen werden erst dann gefunden, wenn das Programm ausgeführt wird, und die Ausführung die entsprechende Zeile erreicht hat.

Hier sind also die möglichen Fehlermeldungen und was sie bedeuten :

?CAN'T CONTINUE ERROR

Sie haben versucht, mit dem CONT-Befehl mit einem Programm weiterzumachen, das entweder nie existierte, oder in dem ein Fehler auftrat, oder das zwischenzeitlich aufbereiteit wurde, wobei Zeilen vom Programm gelöscht oder zum Programm hinzugefügt wurden.

?DIVISION BY ZERO ERROR

Sie dürfen nicht durch Null dividieren, und der Computer darf es auch nicht.

?FORMULA TOO COMPLEX ERROR

Sie haben versucht, einen weiteren IF Befehl auf das THEN eines IFs folgen zu lassen. Das darf man nicht.

?ILLEGAL DIRECT ERROR

Man kann nicht INPUT, DEF FN, GET oder DATA unmittelbar ausführen lassen. Diese Befehle dürfen nur als Teil eines Programms gegeben werden.

?ILLEGAL QUANTITY ERROR

Die Parameterwerte, die Sie einer mathematischen oder Zeichenkettenfunktion als Argument vermittelten, liegen außerhalb des verwertbaren Bereichs. Im einzelnen könnte vorliegen :

a) Ein negativer Tabellenindex (etwa A (– 1) = 0)
b) Ein LOG Aufruf mit Null oder negativem Argument.
c) Ein SQR Aufruf mit negativem Argument.
d) A ^ B wobei A negativ oder B nicht ganzzahlig ist.
e) Verwendung von MID$, LEFT$, RIGHT$, WAIT, PEEK, POKE, TAB, SPC, ON . . . GOTO oder einer Grafikfunktion mit einem falschen Argument.

?NEXT WITHOUT FOR ERROR

Die folgenden Möglichkeiten bestehen: (1) Die Variable des NEXT Befehls ist eine andere als die Variable der FOR Schleife, die als nächste abgeschlossen wird. (2) Der NEXT Befehl enthält keine Variable. (3) Zu dem NEXT Befehl gibt es keinen zugehörigen FOR Befehl, der noch nicht abgeschlossen ist.

?OUT OF DATA ERROR

Ein READ-Befehl wurde ausgeführt, aber alle Daten sind schon vorher eingelesen worden, und es gibt keine weiteren Daten, die dieses READ lesen könnte. Entweder versucht Ihr Programm, zu viel zu lesen, oder Sie müssen dem Ablauf mehr Daten beifügen.

?OUT OF MEMORY ERROR

Jeder der folgenden Sachverhalte erzeugt diese Nachricht: Ihr Programm ist zu groß geworden; Sie haben zu viele Variablen; Ihre FOR Schleifen sind tiefer als 10 geschachtelt; Sie haben mehr als 24 GOSUB Befehle, zu denen noch kein RETURN ausgeführt worden ist; Sie haben einen Ausdruck geschrieben, der dem Apple zu kompliziert ist; Ihre Klammern sind tiefer als 36 ineinander geschachtelt; Sie haben versucht, LOMEM einen Wert zuzuweisen, der niedriger als der gegenwärtige oder zu hoch ist; Sie haben versucht, HIMEM einen Wert zuzuweisen der zu niedrig ist.

?OVERFLOW ERROR

Das Ergebnis einer Rechnung ist zu groß, und der Apple kann es nicht als Zahl im gültigen Bereich repräsentieren. Wenn die Zahl im Betrag zu klein, jedoch von Null verschieden ist, wird sie auf Null abgeändert, und keine Fehlermeldung wird gegeben.

?REDIM'D ARRAY ERROR

Nachdem Sie eine Tabelle schon einmal in ihrer Größe vereinbart haben, versuchen Sie es mit einem zweiten DIM Befehl erneut. Dieser Fehler tritt häufig dann auf, wenn die Standardvereinbarung für die Tabelle dadurch gilt, daß vor Ausführung des DIM Befehls auf ein Tabellenelement zugegriffen wurde, zum Beispiel, wenn Sie zuerst „A (I) = 3" ausführen, und dann später „DIM A (100)". Dieser Fehler kann nützlich sein, wenn Sie nach einem bestimmten DIM Befehl suchen: Fügen Sie am Programmanfang für dieselbe Tabelle einen weiteren DIM Befehl hinzu, lassen Sie das Programm laufen. Nun wird die Fehlermeldung Ihnen die Stelle des anderen DIM Befehls zeigen.

?RETURN WITHOUT GOSUB ERROR

Ein RETURN Befehl wurde ausgeführt, zu dem kein GOSUB vorher ausgeführt wurde.

? STRING TOO LONG ERROR

Sie haben (etwa durch Verknüpfung) versucht, eine Zeichenkette zu bilden, die mehr als 255 Zeichen lang ist.

?BAD SUBSCRIPT ERROR

Es wurde versucht, auf ein Tabellenelement mit einem Index zuzugreifen, der außerhalb des vereinbarten Bereichs liegt. Der Fehler kann auch auftreten, wenn die Anzahl der Indexpositionen (die Dimension der Tabelle) im Zugriff unrichtig ist. Beispiel : LET A (11) = Z mit zugehöriger Vereinbarung DIM A (2).

?SYNTAX ERROR

Es kann sich um überflüssige Klammern, illegale Zeichen, falsche Punktuation usw. handeln.

?TYPE MISMATCH ERROR

Die Linksseite eines Zuweisungsbefehls ist eine numerische Variable, während auf der rechten Seite ein Zeichenkettenwert berechnet wird (oder umgekehrt). Oder eine Funktion, die eine Zeichenkette als Argument erwartet, hat einen numerischen Wert als Argument (oder umgekehrt).

?UNDEF'D STATEMENT ERROR

Mit GOTO, GOSUB oder THEN versuchten Sie, zu einer Zeile im Programm zu verzweigen, die es überhaupt nicht gibt.

?UNDEF'D FUNCTION ERROR

Eine vom Benutzer zu definierende Funktion wurde aufgerufen, die allerdings nie definiert wurde.

ANHANG F
Das ROM des alten Monitors

In diesem Buch haben wir meistens angenommen, daß Ihr Apple ein selbststartendes ROM hat, das heißt, einen Monitor, der sogleich BASIC beim Einschalten des Geräts startet. Wenn beim Einschalten des Apple der Bildschirm sogleich gelöscht wird und die Nachricht

APPLE][

in der oberen Zeile erscheint (und, falls Sie die Diskettenversion haben, das Diskettenbetriebssystem gestartet wird), dann haben Sie ein sogenanntes Autostart ROM.



Bei einigen älteren Geräten muß zuerst die RESET-Taste gedrückt werden, bevor die obige Nachricht geschrieben wird.

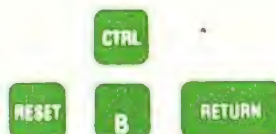
Es gibt aber auch einige Geräte, die einen Monitor-ROM haben, der nicht so funktioniert. Sie erkennen das wie folgt: Wenn Sie Ihr Gerät einschalten, und auf dem Bildschirm erscheinen einige Menge unsinniger Zeichen, die nicht gelöscht werden, dann enthält Ihr Gerät die ROM des alten Monitors, und Sie sollten sich diesen Abschnitt gut durchlesen.

Wie die ROM des alten Monitors benutzt wird

Jedesmal, wenn Sie den Apple einschalten, werden Sie einen Stern (*) als Bereitschaftszeichen links unten auf dem Bildschirm sehen, gefolgt von dem blinkenden Cursor. Daran erkennt man, daß sich das Gerät im Monitorprogramm befindet, das von fortgeschrittenen Programmierern benutzt wird, wenn sie in „Maschinensprache" programmieren. Um auf BASIC umzustellen, müssen Sie die folgenden Schritte ausführen:

1. Sie drücken die RESET -Taste (oben rechts auf der Tastatur).
2. Sie drücken gleichzeitig die Tasten CTRL und B.
3. Sie drücken nun die RETURN -Taste (unten in der Mitte der Tastatur).

In unserem System, Tastenkombinationen und Tastenfolgen darzustellen, können wir das so schreiben:



Wie man sich von versehentlichem RESET rettet

Wenn Sie die Autostart-ROM in Ihrem Apple haben, so stellt ein versehentlicher Druck der RESET -Taste kein Problem dar, denn es bringt Sie zurück in das BASIC-System, das Sie ja gerade benutzten.

Wenn Sie allerdings die ROM des alten Monitors haben, dann versetzt Sie versehentliches Drücken der RESET -Taste ins Monitor-Programm, und Sie werden plötzlich den Stern als Bereitschaftszeichen sehen, gefogt vom blinkenden Cursor. Um zum BASIC System zurückzukehren, müssen Sie eine der folgenden Prozeduren anwenden :

1. Sie haben keine Diskettenversion

Wenn Sie Integer BASIC benutzen oder die Firmwareversion des Applesoft haben (siehe auch Anhang D), dann kommen Sie nach einem versehentlichen RESET durch die Kombination



in das BASIC System zurück. Sie erhalten hiermit wieder das BASIC System, das Sie gerade vorher benutzten, und Ihr Programm ist nicht verloren.

Wenn Sie die Bandversion des Applesoft benutzen und versehentlich die RESET -Taste drücken, so müssen Sie die folgende Kombination benutzen, um BASIC und Ihr Programm wieder zu erhalten.

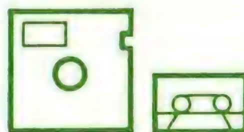


2. Sie benutzen die Diskettenversion

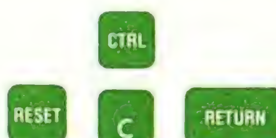
Wenn Ihr Diskettenbetriebssystem (DOS) läuft, und Sie drücken versehentlich die RESET -Taste, so müssen Sie immer



drücken, ganz gleich welche BASIC-Version Sie benutzen. Damit kommen Sie zurück zum DOS und BASIC, und Ihr Programm ist nicht verloren.



Wenn Sie Band- oder Diskettenversion des Applesoft benutzen, wird durch die Kombination



veranlaßt, Sie ins Integer BASIC zurück zu versetzen. Dabei kann das Applesoft-Programm und ein von Ihnen in den Speicher gebrachtes Programm sehr wohl gelöscht werden. Außerdem wird das Integer BASIC nicht richtig initialisiert und wird daher also nicht richtig arbeiten.

3. Sie haben das Applesoft II BASIC Programmierhandbuch

Im Programmierhandbuch für Applesoft II BASIC finden Sie erheblich mehr Einzelheiten des Applesoft, als diese Einführung zu vermitteln sucht. Allerdings wurde das Handbuch in der Annahme geschrieben, daß Sie die Firmware-Karte in Ihrem Gerät installiert haben (siehe auch Anhang D auf Seite 107) und **keine** Diskettenversion verwenden. Daher müssen Sie an jeder Stelle, an der das Handbuch Ihnen die Folge



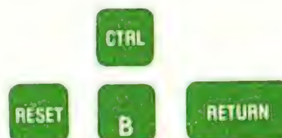
angibt, statt dessen die Folge



verwenden, wenn Sie die Diskettenversion des Applesoft haben, während Sie bei der Bandversion die Folge



benutzen müssen. An jeder Stelle, an der das Handbuch die Folge



angibt, können Sie bei der Diskettenversion des Applesoft zwar dieselbe Folge benutzen, aber Sie werden dadurch in Integer BASIC versetzt, müssen also die Diskette erneut starten (PR #6) und dann das Applesoft erneut von der Diskette laden (FP). Wenn Sie nun die Bandversion des Applesoft haben, werden Sie sich nach der obigen Folge ebenfalls im Integer BASIC wiederfinden. Sie müssen daher auch dann das Applesoft wieder neu laden.

Sachwortverzeichnis für die Einführung ins Applesoft

Die in Klammern angegebenen Zahlen sind die entsprechenden Seiten im Applesoft II BASIC Programmierhandbuch.

R

S

T

U

V

X

Y

Z